Reconstruction of Overhead Electrical Infrastructure for Dynamic Line Rating using Vehicle-mounted LiDAR

A thesis
submitted in partial fulfilment
of the requirements for the Degree
of
Doctor of Philosophy
in the
University of Canterbury
by
Josh McCulloch

University of Canterbury
2018

Abstract

Dynamic Line Rating allows for better utilisation of overhead electrical infrastructure by dynamically determining the current carrying capacity in real-time. It is implemented by combining measurements of the environmental effects with a known model of the conductors. These models can be automatically generated from surveys collected using LiDAR on aircraft. But this approach is expensive and requires conductors to be clear of nearby objects and well easily separable from each other; restricting the application of this technology from use in urban environments. To facilitate the automated reconstruction of conductors in these challenging environments a new method using a structured search anchored to utility poles is proposed. Two core concepts to recover conductors are developed, sag-compensation, to remove the non-linear sag from the conductors before clustering, and a 3D to 2D projection, to increase conductor point density and simplify the clustering phase. These concepts are an improvement over the previous state of the art, which have first classified individual conductor points before performing conductor recovery. It is this novelty which allows the performance previously only achieved on high tension lines in sparse environments, to now be possible in the more dense and cluttered urban setting. Rather than require surveys to be conducted using costly aerial platforms, the proposed method is designed to work with the limitations of data collected from road-bound vehicles. A low-cost LiDAR based system for surveying infrastructure is presented alongside the conductor recovery method. With this approach, overhead electrical infrastructure in urban environments can be more accurately and rapidly surveyed like their high tension counterparts at a fraction of the cost, lowering the barrier to deployment and resources to maintain dynamically rated infrastructure.

Table of Contents

Acknowledgments

While the words within this thesis may be mine, getting them here was a team effort. A big thanks to my supervisor Richard Green, while your insight and experience been invaluable, it has been your encouragement and support that I have most appreciated. To the team at Unison, you have gone above and beyond to make resources available, and offer industry expertise to me. In particular, I would like to thank Thahirah, Kalid, Edwin, and Nu'man. Lastly, Mum and Dad, thanks can hardly do justice for the amount of gratitude I have for you. When I needed the support to get this across the line, you were there. Mum, I still can't believe you read the whole thing, thank you :)

Chapter I

Introduction

It is hard to state just how crucial the reliable delivery of electricity to our places of home, work, and leisure is to the functioning of our society. Like all things, our demand for electricity changes with time, whether it be between morning and night, or on timescales spanning decades. The electrical infrastructure we rely on needs to be designed with these requirements in mind.

Each circuit within an electrical network has a static rating which states the maximum capacity under normal operating conditions. These current ratings are constrained by either safety codes which restrict the maximum amount of conductor sag due to thermal expansion before a clearance violation occurs [1], or the point at which the conductor will begin to anneal, causing permanent damage [2]. A line rating is calculated first when the circuit is installed, but only adjusted infrequently afterwards. Because of the large periods of time between rating adjustments, the ratings are conservative and based on the worst case conditions along with allowing for some deterioration of the network.

In the last 30 years, owners and operators of transmission and distribution networks have begun to determine the capacity of individual circuits and adjust their ratings in real-time using a process called Dynamic Line Rating (DLR). Dynamic ratings are determined by understanding the thermal relationship between the conductor's intrinsic characteristics, electrical load, and the ambient conditions [3]. By implementing DLR, the operators of electrical networks can achieve a capacity of 130% when compared with an equivalent static rating for 90% of the year [4].

DLR can be implemented by directly measuring the sag of the conductors and calculating the amount of residual capacity before a clearance violation

Operating Conditions	Change in Conditions	Impact on Capacity
Ambient temperature	2°C decrease	+2%
	10°C decrease	+11%
Solar radiation	Cloud shadowing	+/- a few percent
	Total eclipse	+18%
Wind	1 meter/s increase, 45°angle	+35%
	1 meter/s increase, 90°angle	+44%

Table 1.1: The relationship between environmental effects and their impact on conductor capacity. Sourced from U.S Dept of Energy [3]

occurs. Alternatively, an indirect approach may be used where the environmental effects are measured in real-time and used alongside a computer model of the conductors to estimate the remaining capacity [5]. To directly measure conductor sag, sensors can be mounted along the conductor's span to measure displacement using Differential-GPS [6], inertial sensors [7], or targets tracked using cameras [5]. Non-contact sensors may also be used to monitor the conductor by measuring the emitted Electromagnetic Field (EMF). The indirect approach to span sag estimation uses sensors to measure weather along the circuit and combines this data with the known load and physical characteristics of the line. Both of these approaches require accurate models of the conductors' physical characteristics. It is determining these characteristics that the research contained within this thesis is concerned with. Once installed, conductors will tend to elongate and sag with time. The sag of the conductors will also change if the poles settle. For these reasons, conductor models need to have additional margins to allow for the variability in how the conductors age [8]. By reducing the cost of surveying overhead conductors through automation, the interval between resurveying circuits can be reduced allowing for safety margins to be more precisely defined. It can also allow for DLR to be implemented in areas of electrical networks where it may previously not have been cost-effective to do so.

1.1 Objective

To develop a fast and low-cost solution to determining the conductor parameters.

Traditionally the survey of conductor profiles has been carried out from aircraft using LiDAR [9] or from the ground by taking manual measurements using laser [10] or acoustic rangefinders [11]. Using aircraft is the most cost-effective way to survey large portions of the network, but requires a significant investment on the part of the operator and is less economically viable to execute on smaller scales. To survey or re-survey smaller portions of the network the conductors are often measured manually on foot; a process that is significantly slower than its airborne counterpart.

The objective of this research is to develop a method of surveying conductors which strikes a balance between the slow manual survey and the more expensive airborne survey. By constructing a system to survey the conductors from a moving vehicle, data can be collected at a significantly higher rate than manually sighting each conductor to complete a measurement. The nature of such a system will result in a large amount of data being collected which is not relevant to the goal of conductor parametrisation. This system will require a method of extracting and reconstructing the conductors from the raw dataset.

1.2 Contributions

This thesis presents three significant contributions to the fields of Dynamic Line Rating and Remote Sensing.

1. The first of these contributions is a method of pole recovery in urban environments which extends upon the existing approaches. This method focuses on robustly localising pole tops; a requirement for the conductor recovery methods proposed in the latter half of this thesis. This method has been presented at two peer-reviewed conferences [12, 13].

2. The second contribution is a novel method of projecting an entire span

consisting of multiple conductors into a 2-Dimensional space for clustering. This approach to clustering conductors in 2-Dimensional space is employed by all three of the conductor recovery methods presented in this thesis. This clustering concept has been published at a peer-reviewed conference [14]

3. The third contribution is a method of determining the layout of the conductors and estimating their sag prior to clustering. This knowledge decreases the complexity of clustering conductors and increases the overall conductor recovery performance. This method of preprocessing conductors has been submitted to a journal.

1.3 Thesis Organisation

The main contributions of this thesis are presented in chapters 5-8 while the earlier chapters 2-4 cover prior work and provide information regarding experiment setup.

- *Chapter 2: Foundational Work*

  This chapter contains an overview of previous attempts to recover conductors from scans collected using both airborne and terrestrial LiDAR. Comparisons are drawn between methodologies, types of conductors, and target environments. Also included is a review of various approaches to reconstructing utility poles in urban environments.

- *Chapter 3: Data Acquisition System*

  This chapter chronicles the development of the Data Acquisition System used to collect the test data used throughout this research. It contains discussions around various design decisions and comparisons between different sensor technologies.

- *Chapter 4: Pipeline*

  To develop the recovery methods discussed in later chapters a framework which manages data and classifiers was developed. This chapter

contains a detailed explanation of how this framework operates, the design decisions involved in creating it, and the other software components and libraries it works with.

- *Chapter 5: Proposed Method for Utility Pole Recover*

  For the reliable recovery of conductors in urban environments, a robust method of determining utility pole locations was required. This chapter contains the pole recovery method developed along with a comprehensive performance evaluation.

- *Chapter 6: Proposed Method for Conductor Recovery*

  This chapter is the first of three focusing on the recovery of overhead conductors. This chapter outlines much of the terminology and general method of conductor recovery which is expanded upon in later chapters.

- *Chapter 7: Proposed Method for Conductor Clustering using Sag Compensation*

  This chapter presents a method for sag estimation and compensation within a span to address shortcomings in the approach discussed in the previous chapter.

- *Chapter 8: Proposed Method for Conductor Clustering using Multiple Sag Models*

  This chapter contains the ultimate method developed for the recovery of conductors in urban environments. It extends the sag compensation method presented in the previous chapter and overcomes various shortcomings contained within the prior approaches.

- *Chapter 9: Conclusion*

  A review of the presented methods is given along with possible areas for future development.

Chapter II

Background

2.1 Introduction

This chapter gives a review of the current methods of automatically modelling overhead electrical infrastructure. It is divided into two main sections, the first focusing on conductor span recovery, and second on utility pole recovery. Many of the existing methods of conductor recovery do not make use of the utility poles they are suspended from, and thus most of the previous work on electrical networks is contained within the conductor recovery section. However, in the cluttered urban environments of interest to this research, a comprehensive understanding of space around the conductors can be highly beneficial to their recovery. As such, the pole recovery section includes many works not specifically relevant to electrical networks, but also urban surveying and environment classification for autonomous vehicles.

2.1.1 Applications

There are two primary applications for accurate models of overhead electrical infrastructure. The most significant likely being the economic advantage of facilitating the implementation of Dynamic Line Rating (DLR). When an overhead circuit is installed, it is given a capacity rating. This rating is constrained by the clearance around the conductor as the conductor will sag due to thermal expansion caused by current losses and environmental factors. DLR allows for this capacity rating to be adjusted as environmental factors change; such as wind and solar radiation [15]. These surveys can be used to implement DLR after the conductors have been installed, or to reconfigure overhead conductors to achieve a higher capacity rating [16].

The second application of survey data is for generating a contextual un-

derstanding of the environment the conductors are situated in; primarily modelling vegetation encroachment. The processes for using LiDAR data for vegetation management and data accuracy validation are well established [17], albeit manually. Most commonly, helicopters are the survey platform of choice used measuring tree encroachment and corridors [18]. In more recent years LiDAR data has been fused with hyperspectral imagery to classify trees and automate the process of monitoring the surrounding environment [19]. As faster sensors have been developed, this technology has been integrated into fixed-wing aircraft allowing for classification of vegetation over larger areas [20].

2.2 Conductor Recovery

The automated reconstruction of power-line spans is not new to the field of remote sensing; there have been many methods developed. Many of these have focused on reconstructing high tension spans from LiDAR data collected from aircraft[21, 22, 23, 24, 25, 26]. These methods show some common patterns. With the application of voxel-based, density-based, and location-based filters the majority of non-conductor points are removed from the scan. Afterwards, conductor primitives are assigned to the conductor like regions; these regions are usually identified by comparing the magnitude of the three eigenvalues describing the local point distribution. A region growing algorithm may then be used to cluster these primitives into larger groups before either RANSAC or least squares is used to fit a conductor model.

One reason that the bulk of conductor recovery research has been focused on high tension spans is these have large inter-span distances and more clearance from the ground and other objects in the scan. In spans where conductors are packed closely it is challenging to calculate eigenvalues for a single conductor. The method proposed by Jwa et al. worked around this issue by using the Hough Transform first to recover the compass direction of the conductors [27].

It is uncommon for conductor recovery methods to first recover the powerpoles or pylons to constrain the conductor search. However, as pointed out by Guo et al, locating the pylons first is helpful for power line reconstruction

[28]. Guo used the JointBoost classifier to identify regions of the point cloud likely belonging to high tension pylons. Between these pylons, a structured search was conducted which identified conductor primitives. Like the previous methods, a region growing algorithm was used to cluster these primitives followed by RANSAC to fit a conductor model.

Cheng et al [29] developed one of the few previous methods for reconstructing power lines using vehicle-mounted LiDAR. Like many of the other conductor recovery methods Cheng using a bottom-up approach, first applying filters, then reconstructing the conductors from more primitive segments. Unlike previous methods, Cheng's method needed to be more robust to segments of the conductors being unobserved due to obstruction from other objects; something that is uncommon in scans collected from aircraft. Fortunately for Cheng, while portions of the conductors were obscured from view, the conductors were well separated from each other allowing for the computation of eigenvalues; a feature that was used for the reconstruction.

2.2.1 Conductor Points Identification

The most common approach used to recover conductors is to start by first identifying small regions of the point cloud which are conductor-like. These conductor-like primitives are then used to reconstruct the entire conductor span. Building a robust system to define these conductor primitives is key to the performance of the rest of the recovery process; and is not a trivial task. The conductor recovery method proposed by Liang et al. overcame this challenge by requiring the conductor region to be manually delineated [30]. Sidestepping the problem is perhaps a reasonable stop-gap solution. A human can quickly make a crude selection, leaving the computer to extract and refine the conductor models. Of course, the ultimate goal is the complete automation of the conductor recovery process, and the remainder of the discussed methods are designed to accomplish this.

The identification of conductor-like primitives requires descriptive features to be created. Many methods make use of a Digital Elevation Model (DTM)[21, 29, 31] to develop point features; including that proposed by Liang et al. Liang used the height of a point above the DTM along with the

LiDAR return information as features to classify conductor points. The LiDAR return information contains information about the object the laser hit, including the intensity, and order of returns, if multiple objects where generated. Liang's method measures the distance between the first and second return. If they are far apart, it is likely that the point may be a conductor, as conductors generally have a large amount of vertical clearance. However, if the returns are closely packed then the object is more likely to be the canopy of a tree. The Dempster-Shafter method is used to combine these features and determine the point class. It is important to note that methods that make use of the relationships between multiple returns are likely to be heavily dependent on the sensors perspective, i.e methods for airborne LiDAR would be less effective on datasets collected from vehicle-mounted LiDAR.

Many methods ignore any LiDAR return information and instead base their point classification on the local point distribution [29, 31, 22, 27, 28]. One challenge that occurs when classifying points based on the local distribution, is handling when the conductors are close together. Commonly the magnitude of the eigenvalues are compared to identify linear structures within the point cloud, but these can be distorted if multiple conductors are within the calculation window. The method proposed by Mclaughlin et al. made use of an ellipsoid shaped window aligned with the flight path to reduce the probability that multiple conductors would pollute the feature generation [26]. This method exploits the fact that the aircraft likely flies parallel to the conductors.

The two above approaches make use of some knowledge about the scan in addition to the raw point location data. The remaining methods only use point location data. One common theme is to divide the point cloud into smaller cuboid regions. Filters can then be used within these regions to reconstruct small-scale structures, or between these regions for large-scale structures. These filters often make use of a pre-computed DTM to allow point height to be used as a feature [29, 31]. Cheng et al. divided the point cloud into small voxels of 10cm along each side. Voxels within one meter of the DTM or in a region with a low vertical spread were rejected. These filters are used to remove the ground and low lying objects. Any voxels belonging to a group of three or more consecutively stacked vertical voxels were also

rejected, as conductors generally have a very narrow vertical distribution of points. For the remaining voxels, the eigenvalues are computed, and any containing linear structures are retained. These filters are designed to preserve small-scale horizontal linear structures in the point cloud but do not attempt to remove larger scale non-conductor like structures. To filter these larger structures the density of the neighbourhood around each voxel is calculated, and any voxel with too few or too many neighbours is rejected. This density filter is tuned to allow structures spanning in a single direction, like conductors, while removing more dense regions, like the edges of buildings. Melzer et al. used a significantly large cell size of one meter [31]. Using a large cell size can allow for better representation of the target structures within a cell, but with the added difficulty of more complex structures being included within the region, such as multiple conductors. To allow for multiple conductors to co-exist within the same region, Melzer substitutes calculating the regions eigenvalues for the Hough Transform, which can find multiple linear structures. A more extreme method is that proposed by Jwa et al. which uses a cell size of five meters [22]. Due to the large cell size, Jwa's method relies entirely on within cell filtering. Features created from a 2-Dimensional Hough Transform, region eigenvalues, and point density analysis are compared against those observed in a training set and cells are rejected based on the results.

Each of the methods discussed above build up conductor primitives using locally derived features. With the application of filters over various scales, points belonging to conductors can be preserved, while non-conductor points are suppressed. However, these methods still result in some non-conductor points being falsely classified as belonging to conductors. The method proposed by Guo et al. attempts to reduce the frequency of these false positives by building a structural understanding of the scan [28]. Using the JointBoost classifier with 26 features derived from geometry and Laser return information, points within the scan are classified as either conductor, pylon, or foliage. During later processing where the conductor points are combined to reconstruct entire spans, the processing is limited to regions between pylons. By first identifying the pylons, not only can the conductor search space be reduced, but the conductor orientation can also be inferred. This knowledge

regarding the conductor configuration is incorporated into the subsequent steps of conductor reconstruction.

2.2.2 Conductor Modelling

The ultimate objective of all conductor recovery methods is to assign a parametric model to the span. These models all assume that each span is confined to a single plane vertical plane [26]; as if the conductor is hanging directly down and not at an angle as if blown by the wind. One of two different models is then used to describe the shape of the conductor on the plane, otherwise referred to as the conductor sag. While the physically accurate model to describe a hanging conductor is the catenary, a parabolic model is often substituted as the catenary equation uses a transcendental function which can make model fitting more difficult [21, 30]. At the scale and tension conductors are expected to be under, the use of a parabolic model over a catenary model is considered an acceptable compromise [29].

The vertical plane which the conductor lies on may explicitly be defined as is the case with the method proposed by Melzer et al. [31] or refined iterative alongside the sag model. The method proposed by Melzer uses a Hough Transform to orient and localise the vertical plane. This plane can then be used to segment the points before the conductor is recovered by fitting a catenary model using RANSAC. Another method for recovery of the conductor direction was proposed by Jwa et al.[27] which uses a Compass Line Filter. The Compass Line Filter generates eight direction hypotheses equally distributed around the compass, then estimates the span direction by choosing the hypothesis with the lowest residual [32]. Jwa's method then goes on to estimate the sag model via a piece-wise process which extends upon the base hypothesis. As new regions are added to the extremes of the segmented conductor the model is refined by incorporating the new points using least-squares. Jwa et al. later published an extension of their earlier work which no longer used the Compass Line Filter, but instead iteratively solved both the direction and sag models; where the direction was estimated using local eigenvectors [22].

While the two previous methods use only the position of conductor prim-

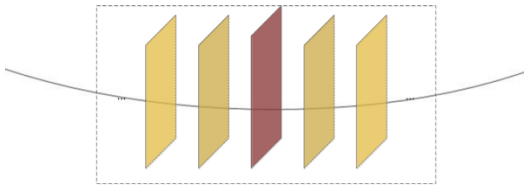


Figure 2.1: Diagram showing the planes perpendicular to the span used for conductor clustering by Guo et al.

itives for model estimation, the method proposed by Mclaughlin et al. also uses the locally computed direction of the conductor primitives to estimated the conductor model [26]. This allows a conductor model to be estimated using only two primitives rather than three. It also does not use a piece-wise growing method, rather opting for primitives being incorporated into the conductor model by consensus; similar to RANSAC.

Each of the conductor model recovery methods discussed above has relied upon the conductor points or primitives derived from them. The method proposed by Guo et al. also uses the conductor points, but unlike these methods, Guo's anchors the search space to the pylons at either end of the conductor span [28]. Once this region between the pylons has been defined, a constrained reconstruction of the conductors is performed. This conductor space between the pylons is divided along its length into cuboid regions. The points within each region are then projected onto a plane perpendicular to the conductor heading; See Figure 2.3. The conductor points on each plane are clustered using a region growing algorithm. By performing clustering at small scales, the curve of the conductors can be ignored and clustering can be done in 2-Dimensions. A descriptor of each clusters' spatial relationship with other clusters on the same plane is generated. The descriptor is used to associated a conductor cluster on one plane with clusters on adjacent planes belonging to the same conductor. This is a region growing approach based on the spatial relationship with other clusters on the same plane, rather than with the same cluster further along the span; common to other methods. Finally, RANSAC is used to fit a catenary model to the segmented conductor points.

2.2.3 Existing Performance

Mclaughlin et al. used the largest dataset for validating their method with 14 km of high voltage lines. Each of the conductors had 0.7-1 meters of horizontal and 5-8 meters of vertical separation. Half of the dataset was used for training the Gaussian Mixture Models for point classification and the other 7 km was used for validation. Their method completely recovered 72% of conductors, and a further 15% where the conductors were either partially recovered or split into multiple models [26]. Guo et al. was able to improve upon Mclanglin's work and push the performance to correctly recover 90% of the conductors on high tension lines [28].

The method presented by Jwa et al. in 2016 was tested on seven spans with varying levels of complexity [22]. One span of interest had elements comparable to those found in urban environments; low and close to foliage. Jwa's methods correctly recovered between 60% and 90% of conductors, with 80% correctly recovered on the urban-like span.

By far the most comprehensive study in an urban environment was completed by Cheng et al. The method proposed by Cheng was tested along 800 meters of powerlines consisting of 6 conductors on three levels [29]. To date, this is the highest performance seen with 94% of the conductors' spans recovered.

It is important to look beyond the presented performance when interpreting the effectiveness of a particular approach. While the method presented by Cheng achieved a very respectable 94%, the conductors are very homogeneous, running parallel and in a consistent layout. These conductor recovery methods can have unintentional biases which can be masked unless tested on a diverse set of data. In the method presented by Jwa above, only seven spans were tested. There was a large amount of diversity between these spans, but the small number of spans makes it hard to have a high level of confidence in the performance of the proposed method. A point that is further reinforced by the fact that the performance varied from 60% to 90% depending on the span tested.

2.3 Pole Recovery

Previous methods for classifying roadside furniture using LiDAR typically use a form of hierarchical classification to overcome the challenges associated with dealing with large variances in point density. One approach is to divide the scan up into large 3-Dimensional cuboid cells and then use various rules to grow regions as proposed by Teo et al. [33]. The method proposed by Teo uses flattened cells in lower portions of the point cloud which can only grow vertically, and smaller cube cells above which can grow to include any adjacent occupied cells. These rules allow for objects with larger volume higher up, e.g. trees, to be captured, while preventing all objects from growing into one region via the ground plane. One significant drawback of using larger cells is the need to split objects that lie in the same base cell using a post-processing step. Alternatively, Cabo et al. proposed a method which uses significantly smaller occupancy cells and then searches for specific structures within them [34]. This eliminates the need for splitting objects in a post-processing step, but is less generic as it requires a heuristic understanding of the objects of interest.

Instead of using occupancy grids to produce a more uniform density for processing, another common approach is to project 3-Dimensional points onto 2-Dimensional surfaces. This projection step increases the point density and can reduce the complexity of pattern matching. Bienert el. al used this method to project tree trunks onto flat surfaces where they appear as circular cross-sections [35]. In this 2-Dimensional space, Bienert can filter the noise produced by small branches and identify the points belonging to the main trunk. McCulloch and Green applied this approach to identify pole-like objects in sub-urban environments [12]. While projecting onto 2-Dimensional surfaces can reduce the complexity of signal processing, there is inherent information loss and distortion. This distortion is especially apparent on objects not aligned with the direction of projection; for instance poles at exaggerated lean angles.

One issue with working with LiDAR data collected with a moving vehicle is the non-uniform density of points within a point cloud; this can be caused by occlusion by foreground objects [36] and the varying angles of incidence

[37]. To improve the point density, we use a method of slicing and projecting which is commonly used when extracting vertical pole-like structures [38, 39, 12].

This radius based rejection model is similar to the rejection criteria used by Lehtomaki et al. who defined a cylindrical rejection model which is applied in 3-Dimensional space [39]

2.3.1 Point-based Pole Classification

Point-based pole recovery is perhaps the most simple of the pole recovery methods. Unlike many other approaches to pole recovery, point-based recovery does not build up a structural representation of the pole, instead only labelling the points. The method proposed by Munoz et al. in 2008 did exactly this; local features were generated for each point, and pole classification was achieved using a Directed Associative Markov Network. A similar approach was implemented by Guo et al. to recover lattice style pylons before performing conductor recovery [28]. Guo's method is further discussed in the conductor recovery section above. Local features typically include point location, surface normal, principal component analysis, and density. The method proposed by Mallet et al. extended upon these standard geometric features to include additional data provided by the LiDAR, including the number of echoes in the waveform and the pulse amplitude [40]. With these additional features, Mallet was able to distinguish reflective objects such as signs and sparse objects such as tree canopies.

The ultimate goal, when creating features for point classification, is to capture enough information of the surrounding environment to discern the type of object the point belongs to reliably. The area of interest needs to be large enough to include the pole-like attributes, such as cylindrical shape, while not be so large as to include irrelevant information like whether the grass around the pole has been cut. While a classifier can be trained to ignore irrelevant information, increasing the signal to noise ratio will be beneficial to performance. In the previous methods surface normal, and Principal Component Analysis were two of the primary features used. These two features both take the local distribution of points and distil them down into a mini-

mal set of vectors. This works well if the point distribution is homogeneous within the area of interest, but can tend to mask smaller scale structures if they are present. To better describe the point distribution, Behley et al. used spin images as features for their classifier [41]. Spin images take the form of a 2-Dimensional histogram which is swept around some axis centred on the point of interest. As the histogram is rotated, points will pass through the cells of the image and these are counted. Often the axis of rotation is the computed surface normal. However, the computed surface normal on poles within a LiDAR scan can be unstable due to the high curvature and low point density. Behley overcame this by instead rotating around the vertical axis. One of the main outcomes of this research was that the resolution of the spin images had little effect on the performance of the classifier. Instead, the performance was more dependent on the radius of the histogram.

2.3.2 Pole Classification using Voxels

Depending on the density of the data within a point cloud, the individual points may not contain a significant amount of information; they will also be subject to some amount of noise generated during the measurement process. One method to overcome these challenges is to combine the points into some larger, albeit still primitive element. These primitive elements are generally called Voxels, a portmanteau of "volume" and "element", analogous to pixels in 2-Dimensional images. These voxels are generally aligned on a 3-Dimensional grid [34, 33] and can be as primitive as defining whether a cell is occupied, or contain significantly more information, such as density, colour, and surface normal.

An approach proposed by Aijazi et al. breaks the grid-based voxel trend. These voxels referred too as super voxels are created through an iterative algorithm. An unexplored point is selected, and a super voxel is created from it and its neighbouring points. While these super voxels are axis aligned cuboids, they vary in size to include all of the neighbouring points, See Figure 2.2. These super voxels include features which describe the surface normal and laser return information. Objects are then clustered using a Euclidean clustering algorithm and classified based on a predefined set of thresholds.

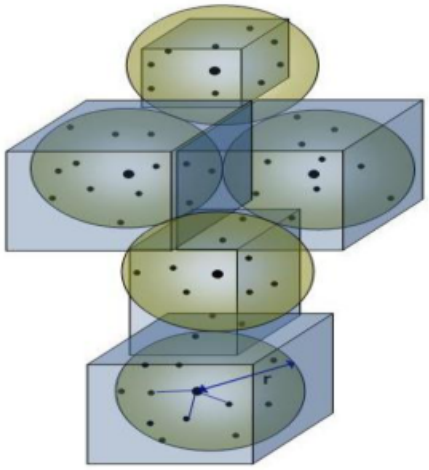


Figure 2.2: Example of the floating voxels used by Aijazi et al.

Using this method, Aijazi was able to recover 90% of poles within the dataset.

A more traditional application of voxels is seen in the method proposed by Cabo et al. Cabo had two reasons for the use of voxels, firstly density normalisation, and secondly data compression. Voxels can work exceptionally well for both of these, essentially downsampling regions of high density while preserving structure at the designated scale across the entire dataset. Cabo used a voxel size of 10 cm. To recover the poles from the voxel dataset 2-Dimensional analysis was first carried out on each horizontal layer. Within each layer connected components were clustered. The clusters were then rejected if they were larger than the expected cross-section of a pole or they did not include enough horizontal clearance with the closest neighbouring cluster. After processing each of the layers within the voxel dataset, clusters aligned vertically between multiple layers were combined. This method was able to segment up to 94% of the poles within the dataset.

In the two previous methods, voxels have conformed to match the underlying data. However, voxels can also be used to enforce a particular structure and enhance objects with the desired characteristics within the point cloud. An example of this can be seen in the method proposed by Teo et al. Teo's approach was to use flattened voxels for regions less than 3 meters above the ground, and cube-shaped voxels for higher regions; See Figure 2.3. The cube

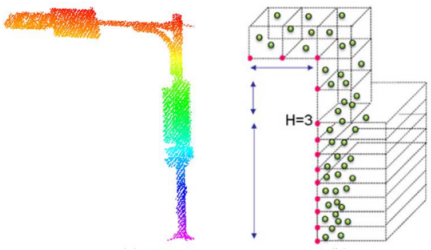


Figure 2.3: Example of the two distinct voxel sizes used by Teo et al.

voxels had sides with a length of 1 meter while the flattened voxels were half as high and twice as wide. A Euclidean clustering algorithm was then used to extract connected regions from the point cloud. The particularly clever part is the use of 1.5 meters for the clustering distance. This threshold only allows region growing vertically in the flattened voxels below 3 meters while allowing growth in any direction above. The result is a heavy bias towards vertical structures lower down while still allowing more complex structures high up as seen in street signage and utility poles. Teo did use some post-processing to separate objects that may have been co-located within the same flattened voxels. For this, a point level euclidean clustering algorithm with a threshold of 0.15 meters was used. Finally extracted objects are filtered based on their height, position, shape, and cross-sections. Teo et al. achieved the highest performance of the reviewed voxels based pole recovery algorithms with precision and recall of 96%.

2.3.3 Pole Classification using Cross-sectional Slices

The approach of using the pole cross-sections for classification is designed to exploit the constant vertical structure of the pole along its length. While modern approaches use multiple cross-sections derived from a 3-Dimensional point cloud, earlier work by Press et al. attempted to use a single slice from a single beam LiDAR [42]. Press's single beam LiDAR was mounted on an indoor robot which needed to identify objects within its environment to aid in navigation. Press's method searched each complete scan from the LiDAR for the cross-sections of pole-like objects which appeared as arcs. Press was then able to calculate the radius of the pole which was useful for identifying

particular poles within the environment.

Since the work presented by Press, pole classification has moved into 3-Dimensions, and a common approach has emerged. Generally a 3-Dimensional point cloud is first divided into discrete horizontal slices, structures within these slices are then clustered, and finally larger vertical structures are reconstructed from the primitives within each slice [43, 39, 38]. Of course, there is variation in how each of these steps may be executed. For instance, Lehtomaki et al. didn't use horizontal slices from a larger 3-Dimensional point cloud but instead used separate sweeps from different lasers on a single LiDAR unit. This approach allowed for pole classification without first performing 3-Dimensional reconstruction of the environment allowing for real-time operation.

While the above methods use cross-sectional slices for pole segmentation, they can also be used to identify pole type as shown by Pu et al. [44]. Pu first segmented objects within the point cloud by removing the ground and clustering the remaining connected points. The clustered objects were then filtered using a set of heuristics and features, including size, position, shape, and colour. The remaining pole-like objects are then divided into horizontal slices. The horizontal slices are checked against each other to ensure features including shape and size are constantly found throughout them; as would be expected from roadside poles. Unfortunately, this method resulted in a high false positive rate of 60% as many trees were classified as poles.

2.3.4 Pole Classification by Ground Subtraction

Object segmentation by ground subtraction is by far the most common method of segmentation in urban environments and is often used as a first step towards pole classification [45, 46, 47, 48, 49, 50, 51]. The general approach is the same as described above in the method presented by Pu et al. First points belonging to the ground are filtered from the point cloud, then connected regions are clustered using a Euclidean-based clustering algorithm. To filter ground points a height based filter is used in conjunction with a Digital Elevation Model (DEM). While the construction of DEMs a large field, there are two methods commonly used concerning LiDAR in urban en-

vironments; those being iterative plane fitting and cell-based filtering. The cell-based filtering approach generally divides the point cloud up into cells on a horizontal 2-Dimensional grid. Within each of these cells, the computed ground elevation is set to the lowest point. The cell-based DEM computation generates a dense 2-Dimensional dataset that can then be further processed using common image processing techniques such as a close operation to fill areas where the ground was not observed [50], i.e under a house.

With the ground points removed, processing can begin on the remaining structures. These remaining points are often clustered with a Euclidean-based clustering algorithm [51, 45]. An additional pass over the clusters with a K-Nearest Neighbours graph cut algorithm may be executed to separate loosely connected objects which have been falsely joined [46, 47]. A crude filter is usually then applied to remove any structures too large to be of interest, such as buildings. It is after these first steps to segment and cluster roadside objects that the discussed methods of pole recovery begin to diverge.

The points within each cluster can be individually classified, as seen in the method presented by Yokoyama [47] and Tombari [48]; similar to the point-based pole classification discussed above. Both used PCA to generate features for each point. Tombari used an SVM to identify pole points followed by a Markov Random Field (MRF) to cluster the classified points. This method achieved a 75% recall with a precision of 81%.

The alternative to point classification is to classify the cluster in its entirety. The method proposed by Golovinskiy used both shape features, including size and PCA, along with contextual features such as whether the cluster was in line with an other or next to a road. Golovinskiy also compared multiple classifiers and found that the Random Forest achieved the highest precision at the cost of recall, K-Nearest Neighbours and SVM had lower precision and higher recall [45]. The method proposed by Velizhev et al. also used cluster level classification. Like Golovinsky, Velizhev also used height and location, but also used a single large spin image which encompassed the entire object. The classification was then performed by finding the closest matching pre-computed class. A dictionary of examples of each class was created using K-Means on a set of pre-classified poles. This method resulted in an 80% recall with a precision of 69%.

2.4 Conclusions

A large amount of effort has been focused on developing a variety of methods for automatically recovering conductors from airborne LiDAR scans. A lot of focus has been put on high tension conductors while infrastructure in urban environments has been neglected. This could be for a variety of reasons including the assets being of less importance than their high tension counterparts, the more complex, cluttered environments they are located in, and the logistical complexity of operating airborne LiDAR over densely populated areas. In contrast, the automatic recovery of roadside poles using vehicle-mounted LiDAR has been well researched; partially because poles are significantly better represented when scanned from the side rather than from above.

Because of the little amount of research that has been done to recover power-lines in urban environments, there is a large amount of freedom regarding how one could approach this. With the large amount of research that has been invested into pole recovery in urban environments, it would seem that first recovering poles to inform the conductor recovery may be an ideal approach; as touched on by Guo et al. By drawing on these two areas of research, pole recovery, and conductor recovery, the development of a robust solution for reconstructing power-lines in urban environments can be developed.

Chapter III

Data Acquisition System

In this chapter, we cover the development of the Data Acquisition System (DAS) used to generate the point clouds discussed in this research.

3.1 DAS Requirements

3.1.1 DAS Platform

During our initial development of the DAS, we were targeting two possible survey platforms: ground-based vehicle and air-based, through the use of Unmanned Aerial Systems (UAS). A vehicle mounted DAS can be both heavier and bulkier than that of one designed for a UAS, allowing for more freedom when choosing hardware. Meanwhile, an airborne DAS can be a lot more flexible in choosing observation locations and can be used in areas without road access.

The sensors discussed in this section were chosen because they met the requirements in order to be mounted on a UAS: low power and light weight. However, the team responsible for developing the UAS was disbanded and the work on the UAS was shelved. Because of these factors, the DAS was never flown, and all of our data has been collected from the rooftop of a road-bound vehicle.

Towards the end of this research, we discovered that there was a third DAS platform that could be targeted. In areas inaccessible by road vehicles, the DAS could be mounted to linemen walking alongside the power lines. While considerably slower than a vehicle mounted system, it is a lot faster and less mentally taxing than the existing procedure which involves manually measuring the lines with a laser tape measure. This is discussed further in the Future Improvements section below.

3.1.2 DAS Resolution

In this discussion, we have assumed the median distance between the conductors and DAS to be 6.5 meters. This value is calculated by taking the minimum overhead line clearance value of 6 meters [52], assuming that the DAS will be mounted on top of a vehicle at 1.5 meters high, and scanning from the road at a distance of 4 meters. In this scenario, we would expect to see conductors no closer than 6 meters at their apex (lowest point) and increase to at least 7 meters as they approach the poles. These can be considered the minimum resolution requirements. In reality, there can be multiple levels of conductors, with the higher voltages further up the poles; in our later chapters, we find them regularly above 10 meters.

3.2 Sensor Technologies

3.2.1 RGB-D Camera

An RGB-D camera is comprised of two sensors: a standard colour camera, and one of several depth sensing technologies. These sensors are used to capture a colour image and a depth image of the same region. A depth image, like a colour image, is a 2-Dimensional array of pixels, but unlike the colour image, each pixel represents a distance from the camera. These images are normally rectified to account for any distortion caused in the camera's optics, and a linear transform is provided for mapping pixels between the two images. These cameras were popularised in 2010 with the release of the Microsoft Kinect [53], a sensor designed to be used for interacting with their Xbox 360 games console. Since the release of the Kinect, we have seen additional low-cost RGB-D sensors such as the Intel Realsense family of sensors. There are three technologies which are commonly used by these low cost RGB-D sensors; these are stereo-vision [54], structured light [55], and time of flight [56].

RGB-D cameras which use stereo-vision, such as the Intel R200, capture two images from separate cameras; features common to both images are extracted, and the depth of these points can be calculated using stereogrammetry. Stereogrammetry requires the two images to have strong, distinct

features to operate; the Intel R200 ensures such features are present, by projecting a pattern of infrared light into the imaged region. This approach of aiding 3-Dimensional reconstruction is referred to as active stereo.

Structured light based RGB-D cameras project a known pattern of light into the scene and observe how it is distorted. Because the light is projected from a location offset from the camera, objects closer to the sensors will experience higher amounts of distortion due to a larger angle of convergence. The Kinect V1 uses structured light and an infrared camera to capture a depth image. Unlike active stereo, structured light based sensors do not use natural features to recover depth information. This means if the sensor is in an environment where the structured light is unobservable, i.e., in direct sunlight, the performance of the depth sensor will be reduced.

Time-of-Flight (ToF) based sensors measure distances by timing the duration for light to leave the illumination source on the camera, bounce off any objects in the scene, and return to the sensor. The Kinect V2 uses a periodically intensity modulated light source to illuminate the scene. The time taken for the light to return to the sensor induces a phase shift in the observed intensity. This phase shift can be measured by comparing the incoming waveform with that generated by the light source [56]. Greater distances result in larger phase shifts. Unlike stereo-vision and structured light based sensors, the accuracy does not degrade with distance. However, distances resulting in a phase shift of more than one period become ambiguous with their less than one period equivalent, and cannot be measured. For the Kinect V2, the upper range limit imposed by the complete period phase shift is 4.5 meters. One other benefit of ToF based cameras is every pixel on the depth sensor can provide an independent depth measurement, unlike previously discussed sensors which need to identify features spanning multiple pixels.

3.2.2 Testing

At the time of testing the Intel Realsense Cameras were not widely available, and so are not included. Our testing did include the Kinect V1 (structured light) and Kinect V2 (ToF) cameras and involved imaging two roadside structures from the rooftop of a stationary vehicle. The imaged structures were

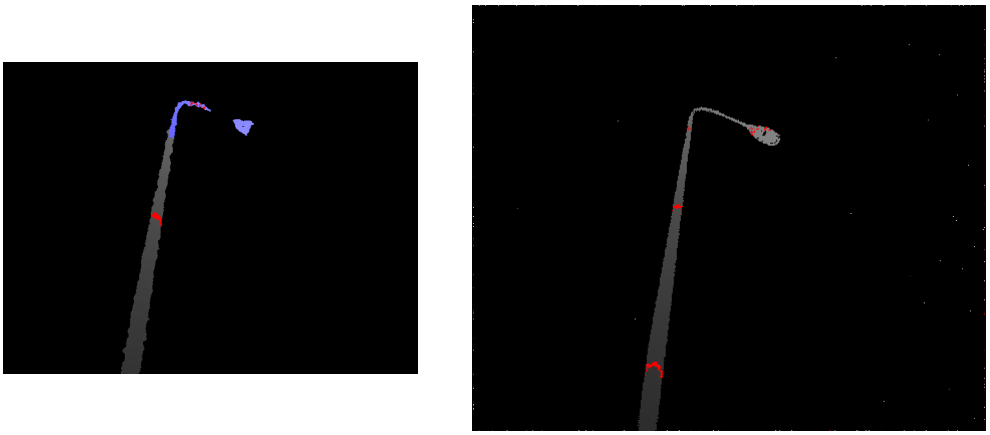


Figure 3.1: A comparison of depth image resolution between Kinect V1 (left) and V2 (right). Distances of 2, 4, and 6 meters are coloured red. Distances beyond 5 meters on the Kinect v1 are reported as having reduced accuracy and have been coloured blue accordingly.

a lamp post (Figure 6.1) and a power pole with multiple conductors, cross-arms, and insulators (Figure 3.2). Due to both sensors using infrared light to measure distances, the tests were conducted after sunset, reducing the amount of ambient infrared light in the scene. This was done to ensure the best case performance of these sensors was measured [57].

Unfortunately, we found neither of the sensors were able to meet our requirements. While larger structures such as the poles were detected (Figure 6.1), the sensors were not able to detect any points on the conductors (Figure 3.2). The Kinect V2 was able to detect thinner structures, such as supports and guide wires, due to its higher resolution depth image. However, the conductors have too small a cross-section to be detected at our target range. Given these results, current commercial RGB-D cameras are not an ideal primary sensor for scanning overhead utility networks.

3.2.3 LiDAR

LiDAR is a remote sensing technology similar to RADAR, but using pulsed laser light instead of radio waves [58]. Like RADAR it measures distances by timing the round-trip time for light to travel from the laser to the object, and back to the sensor. It is tempting to compare LiDAR with the Kinect V2

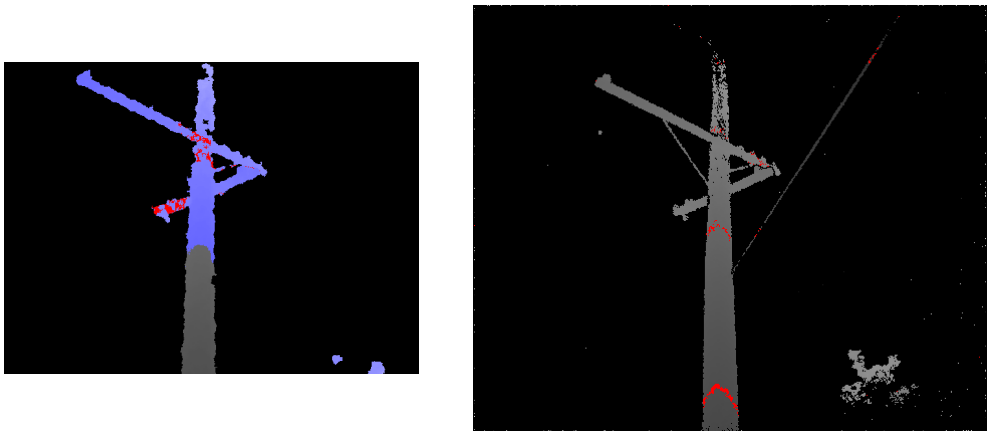


Figure 3.2: Comparison of depth image resolution between Kinect version 1 (left) and 2 (right). Notice that thin structures, such as the cable stay, are visible in the version 2 image. See Figure 6.1 for explanation of colouring.

which also uses ToF to measure distance; there are however some significant differences. The Kinect V2 requires multiple samples in time to measure the phase of the modulated light and calculate the distance. LiDAR uses a single pulse of light, measuring the time from emission to reception. The main operational difference is the rate at which data is captured. The Kinect V2 captures an entire field simultaneously at a resolution of 512x424, and at 30 hertz. Scanning LiDARs, however, do not have the same concept of a frame, they continuously rotate and each revolution blends seamlessly into the next. The fact that each point in a LiDAR's scan was captured at a different time needs to be considered when performing reconstruction operations. Because the LiDAR does not rely on measuring a phase shift like the Kinect, it is able to measure greater distances without running into the ambiguity problem discussed above.

After testing the RGB-D cameras revealed they are not ideal for our application, we acquired a scanning LiDAR: the Velodyne VLP-16 (PUCK). The PUCK is small enough to be mounted easily to the roof of a vehicle or below a quadrotor. The name "PUCK" comes from its resemblance to a hockey puck, see Figure 3.3. It is comprised of 16 lasers, each with a vertical separation of 2° centred on the horizon. This gives a total vertical field of view (FOV) of ±15° which scans around the unit and a nominal rate of 10

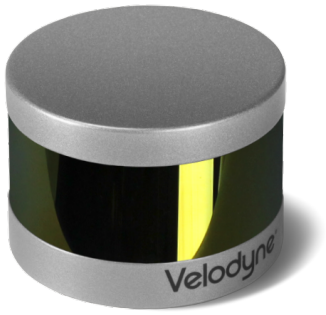


Figure 3.3: The Velodyne VLP-16 (PUCK) LiDAR

Figure 3.4: Navya, an autonomous vehicle using two Velodyne VLP-16 LiDARs for localisation and obstacle detection. Currently undergoing trials at Christchurch International Airport.

revolutions per second [59]. The PUCK has an effective range of 100m and accuracy of ±3cm. Due to the PUCK's small size and comparatively low cost (USD$ 10,000), it is being targeted at applications in the autonomous automotive and UAS fields, see Figure 3.4. The laser pulse of the VLP-16 has a divergence of 0.18 degrees and this is a significant factor in the return signal strength. As the beam's diameter expands beyond the cross-section of the conductor, energy is wasted; only the portion of the beam intersecting the conductor can contribute to the final measurement. Table A in Appendix A contains a set beam to conductor ratios for different LiDAR configurations.

3.3 DAS Construction

For this research we developed two DAS configurations, both use the LiDAR as the primary means of data acquisition and an embedded Linux computer for data recording, but differ in their configuration of supporting sensors and software. We will refer to these two configurations as the Minimal DAS and Complex DAS. The Minimal DAS is our second generation based upon the Complex DAS, and while offering a subset of the functionality of the Complex DAS, it is more reliable and requires less user interaction to prepare for data acquisition.

3.3.1 Hardware

LiDAR Mounting

Choosing a mounting solution for the LiDAR is an exercise in compromise. To optimise laser returns from the conductors the LiDAR should be positioned as close as possible to the conductors, and the laser beams should have a high angle of incidence to them.

The LiDAR has 16 lasers, each makes 1875 samples per rotation; this results in a significantly higher sample rate in the plane of rotation. When choosing an orientation, some consideration should be given to increasing the likelihood that the conductors will cross the paths of all 16 lasers. We also want to decrease the laser/conductor intersection distance; as the distance increases the laser's cross-section increases. At 10 meters the laser beam has a width of more than 4 times that of the conductor; the Puck's laser diverges at an angle of 0.18° [59]. With these considerations in mind, the best configuration would be for the LiDAR's axis of rotation to be parallel to the conductors. This will increase the angle of incidence, decrease intersection range, and increase intersection probability. However, there is one competing consideration.

The algorithms we used for reconstructing the scanned environment require that a large amount of the scene is present between successive scans by the LiDAR. Unfortunately, the configuration previously described for maximising conductor representation in the collected data does not look along

Figure 3.5: Ford Mustang GT using four Velodyne LiDAR sensors mounted in various orientations. Source: bizjournals.com

the direction of travel; instead, the axis of laser rotation is near parallel to it. To maximise inter-scan feature overlap, we would instead prefer to mount the LiDAR in an upright configuration, similar to many autonomous vehicles, see Figures 3.5 and 3.6.

After an investigation where we tested various amounts downwards inclination along the direction of travel, we settled on a value of 15°. At this angle we can expect 14 of the 16 lasers will intercept the conductors within the 100 meters of effective range, with the closest intersection being 8 meters; see Table 3.1 and Table A.1 for extended results. At 15° inclination the highest angled laser will scan forward in the direction of travel, and will be able to see up to 100 meters ahead of the DAS; the lowest laser will behave the same in reverse, allowing for maximum inter-scan overlap. This balance in mounting configuration allows the DAS to both scan the entire height of the poles and obtain a good representation of the conductors, all while providing the maximum inter-scan overlap to provide our reconstruction algorithms with the best possible data.

Complex DAS

The Complex DAS was our first system used in the field to collect data. It is based around the Odroid-C2 embedded Linux computer. Embedded Linux computers are ideal because they provide many of the same I/O as desktop computers, such as Ethernet and USB, but also provide I/O commonly found

Figure 3.6: Google's "self driving" vehicle with a single Velodyne LiDAR mounted horizontally. Source: wired.com

Table 3.1: LiDAR / Conductor intersection characteristics for different levels of LiDAR inclination. These values are based on a vertical separation between the LiDAR and conductors of 4m.

	0 Degrees	5 Degrees	10 Degrees	15 Degrees
Intersections <100m	7	9	12	14
Closest Intersection	15.5	11.7	9.5	8.0
Closest Coverage Ratio	6.10	4.9	4.2	3.8

on micro-controllers such as UARTs and I2C. Also importantly, they can have large amounts of storage available for high-bandwidth sensors, such as the LiDAR.

Connected to the embedded computer we have the LiDAR, GPS, and an IMU, using Ethernet, UART, and I2C respectively, see Figure 3.7. These sensors and computer are packaged into a small plastic container along with a Lithium Polymer (LiPo) battery, see Figure 3.8.

The secondary sensors, GPS and IMU, are low cost off the shelf components comparable to what would be found in most cell phones. The GPS communicates with the computer using serial and is installed in the same enclosure. The IMU needs to be directly attached to the LiDAR for its readings to reflect that of the sensor. The IMU, an MPU-6050, was mounted inside a small 3D-printed case, see Figure 3.9, and then attached to the underside of the LiDAR.

The Complex DAS used the Robot Operating System (ROS) [60] for recording sensor data. ROS provides facilities which makes it ideal for recording sensor data. It uses a message-based form of inter-process communica-

tion. Each sensor can have a dedicated process reading values and making them available to others using these ROS Messages. This can be combined with the application ROS-Bag, which can eavesdrop on these messages and save them to a Bag file. These Bag files maintain all of the information within these messages, including which process created them and at what time. A Bag file can be "replayed" at a later time for analysis as if the sensors were present and in the field. This function can be invaluable when developing methods for reconstructing the scanned environment.

While performance and functionality of the Complex DAS were acceptable, one major flaw was the poor usability and stability of the system. The Robot Operating System requires a large number of additional libraries to support it which can be difficult to compile and maintain in the limited computing capabilities of an embedded computer. The Complex DAS also required a great deal of technical knowledge to operate, including, Linux remote administration and operation, and ability to launch a series of ROS processes (nodes) in the correct sequence. These challenges ultimately led to the development of the Minimal DAS discussed below.

Minimal DAS

The Minimal DAS was designed on what we had learnt from reconstructing various scans based on the data collected by the Complex DAS (Section 3.4). The Minimal DAS does away with ROS and the supporting sensors, IMU and GPS, retaining only the LiDAR. Instead of using ROS Bag to record and later replay the collected data for reconstruction, the Minimal DAS uses TCPDump, a Linux utility, to save raw LiDAR data in the form of UDP packets directly.

The Minimal DAS also switches out the Odroid-C2 for a Raspberry Pi 3 (RPi3) embedded computer. This results in a reduction in the available RAM, from 2GB to 1GB, but without ROS installed this is still adequate. The RPi3 has some advantages which made it an ideal computing platform. It has a much larger community of users and developers resulting in software that is generally better supported than that provided for the Odroid-C2. It also has built in Wi-Fi which allows for the Minimal DAS to be easily

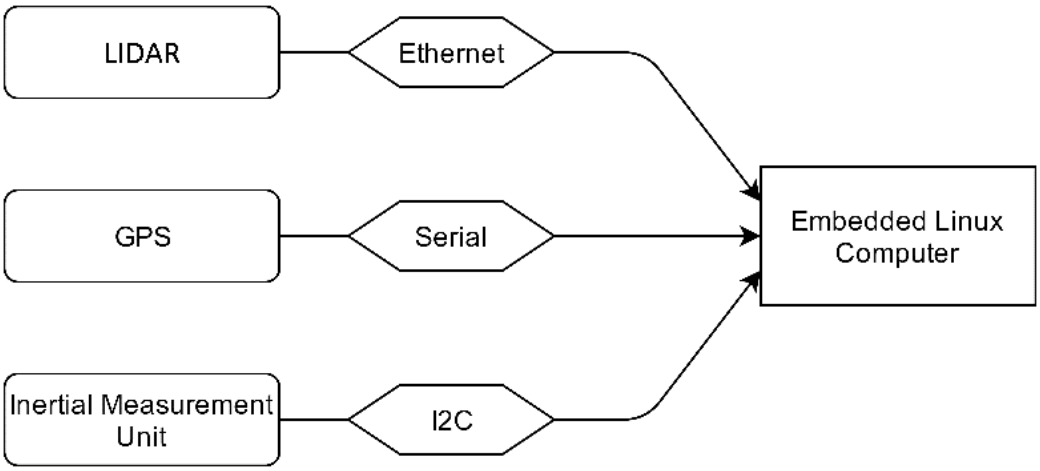


Figure 3.7: Each of the DAS sensors is connected directly to the embedded Linux computer.

administered using the web browser on a cell phone. Lastly, upon completion of a scan the packet capture file (pcap) can be downloaded using the same Wi-Fi network used for control.

3.4 Scan Reconstruction

For reconstructing a scan we used the LiDAR Odometry And Mapping (LOAM) algorithm developed by Zhang et al [61]. The LOAM algorithm can be used to recover the odometry of the DAS and re-project the scanned points into a global coordinate space. LOAM deals with many difficulties which arise when using a LiDAR to generate a dense 3-Dimensional point cloud. LOAM can reconcile the difference between edges of structures and edges of an observable region by applying a heuristic local filter. LOAM stitches together successive LiDAR scans by generating two separate point clouds, corners and planar surfaces, and then uses a variation of the Iterative Closest Point algorithm to align them. LOAM is also able to overcome the distortion present when the LiDAR is being moved while scanning. A single revolution of the LiDAR takes 100ms; when travelling at 50kph this results in 1.4 meter change in viewpoint. To correct this distortion, LOAM uses an iterative process of aligning the point clouds to recover the odometry, and using the derived velocity to partially correct the distortion.

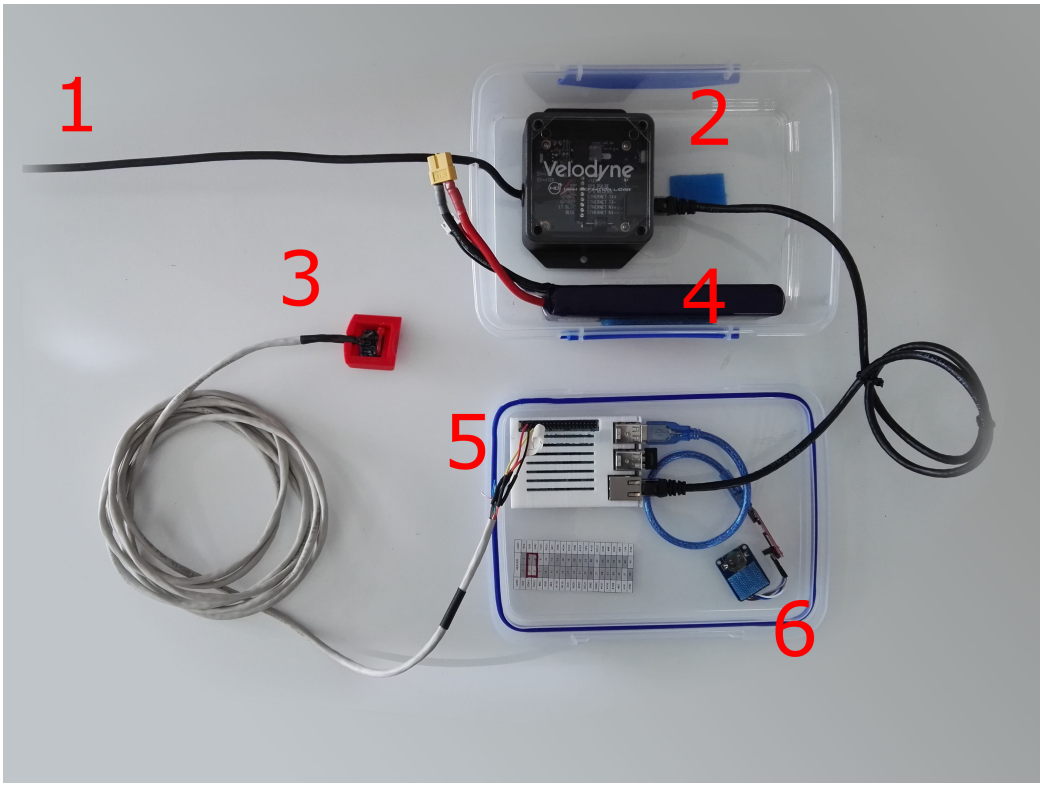


Figure 3.8: Complex DAS Hardware: LiDAR (1), LiDAR breakout (2), IMU (3), LIPO battery (4), Embedded computer (5), GPS (6).

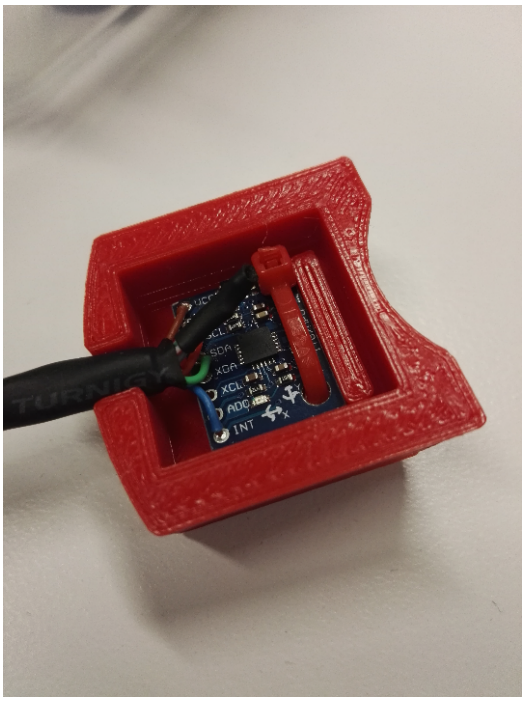


Figure 3.9: The IMU (MPU6050) is fixed inside a custom case and mounted to the bottom of the LiDAR.

Figure 3.10: A side view of a 1.4km straight road. Note the downward distortion which is present when the LiDAR is inclined at higher angles.

Unfortunately, our reconstructions do contain a distortion in the form of a downward pitch approximately along the direction of travel when the LiDAR is positioned at higher inclinations, see Figure 3.10. We measured the total distortion using a 1.8km circuit in an urban environment, see Table 3.2. For smaller inclinations ($\leq 10^{\circ}$) the drift has no preferred direction, however, at 15° the downward pitch is significant. The majority of the datasets used in this thesis are less than 500 meters stretches of road and we take no measure to correct this downward drift. However, before this technology can be rolled out, this shortcoming needs to be addressed, and we have proposed several solutions in Section 3.5.

One major consideration is the representation of the conductors in the

Table 3.2: Reconstruction performance for various LiDAR inclinations over 1.8km

Inclination (Degrees)	Drift (Meters)	Points On Conductors
0	13.3	32
5	10.9	52
10	14.1	158
15	21.1	214

reconstructed scan. The LOAM algorithm does not use the conductors for reconstruction, surfaces and corners are used, but in order for the conductors' structure to be recovered, they need to be present in the final reconstruction. We can see the inclination of the LiDAR does strongly influence the average number of conductor points in the scan (Table 3.2) as predicted in our analysis (Table 3.1). A further discussion of the conductor recovery requirements is contained within Chapter 6.

3.5 Future Development

The most significant problem with the DAS we have developed is the drift that is experienced over longer distances. In our research, we were able to ignore this issue because our algorithms were able to be developed using scans less than 1km in size. However, for this technology to be practically deployed some method of removing this distortion must be incorporated. We experimented with two sensors for augmentation, an IMU and GPS, but did not complete the integration because of time constraints coupled with a lack of requirement for our work. Integrating an IMU would enable the correction of the downward trend our reconstructions were experiencing over longer distances. By measuring the orientation of the sensor the pitch down distortion could be measured and accounted for. An IMU would not be able to account for all drift over longer distances, and so we also suggest the integration of a GPS. With a simple set up using consumer grade GPS an accuracy of 5 meters is achievable [62], and more complex differential systems can go into the sub-decimetre range [63].

Chapter IV

Classification Pipeline

4.1 Overview

The Classification Pipeline discussed in this chapter is the core framework used for managing data and classification methods. In the following sections, we outline the various concepts used to construct the pipeline. We also discuss two classifiers used to recover the ground and foliage from within a point cloud.

4.2 Motivation

During the early development of our classification methods, we found that without a defined structure the code base would quickly become unwieldy. To mitigate this, we set out to develop a pipeline which would allow for various classifiers to be easily chained together, and define a standard set of interfaces to allow for communication between them. We also wanted to be able to assemble non-linear chains of classifiers, as this would allow for a classifier to have dependencies from many previously executed classifiers.

4.3 Construction

The classification pipeline is constructed using three concepts: classifiers, segments, and the pipeline.

4.3.1 Segments

Segments are used to represent knowledge about objects within a point cloud. It is important to note that segments do not have a one to one relationship

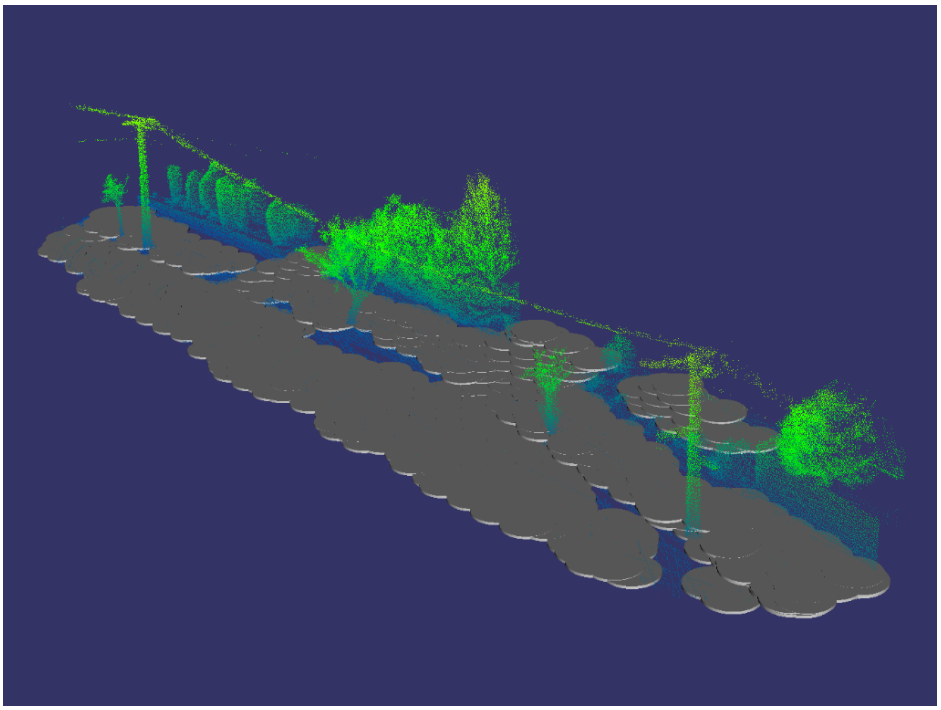


Figure 4.1: Ground segments rendered as short cylinders. The width of the cylinder is determined by the radius of the ground segment.

with objects in the point cloud. Instead, it could represent a sub-component of an object or may represent a region which could contain multiple objects. Segments are created by classifiers and stored in a single collection maintained by the pipeline.

All segments inherit from a common parent segment data type. All segments have a location and a collection of points found in the global point cloud which they represent. On top of this base information a segment will have additional variables defined depending on the represented entity; these are outlined below.

Ground Segment

The ground segment contains a minimal amount of data. Along with the base variables, it only maintains the radius around the location which is known to be ground. Throughout this thesis, ground segments are represented as grey disks as seen in Figure 4.1.

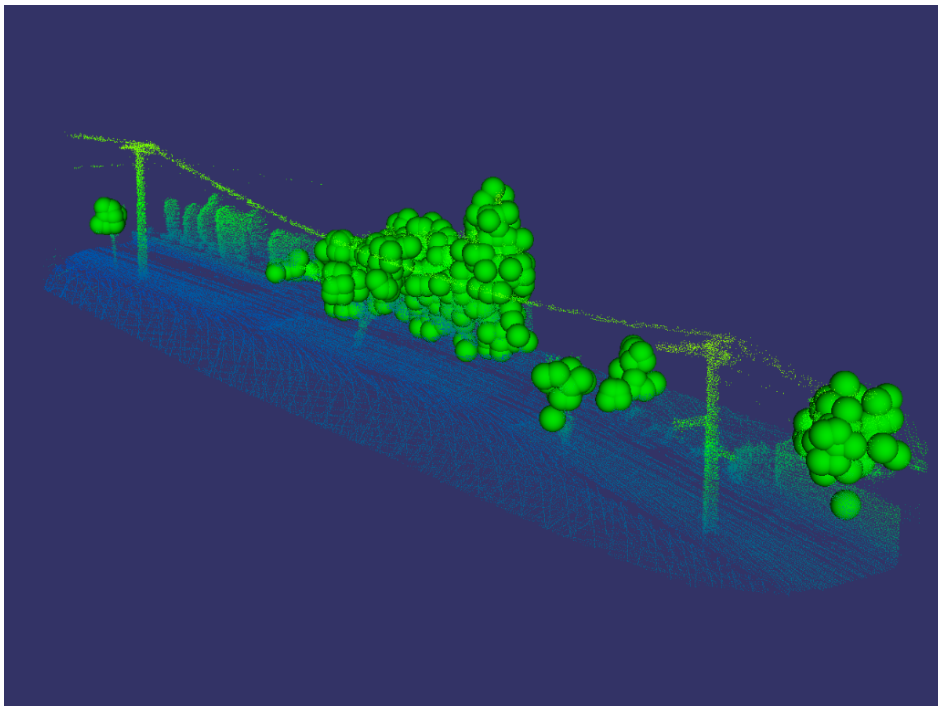


Figure 4.2: Bush segments are rendered as green spheres. The radius of the sphere is determined by the radius of the bush segment.

Bush Segment

A bush segment, like a ground segment, only has a radius variable. This radius represents a spherical region around the segment's location which has been classified as being a bush. Bush segments are not only used for low growing foliage but any plant with bush-like structure such as the canopy of a tree. Bushes are rendered as a green sphere as seen in Figure 4.2.

Pole Slice Segment

The pole slice segment is used to represent regions of the point cloud that are likely to belong to a pole. Like the previously described segments, pole segments also have a radius which is the estimated pole size. The pole slice also has a state variable which is used to indicate to later classifiers whether it was able to be successfully stitched into a pole or if it is awaiting processing. Pole slices are rendered as small vertical cylinders of various colours as shown in Figure 4.3.

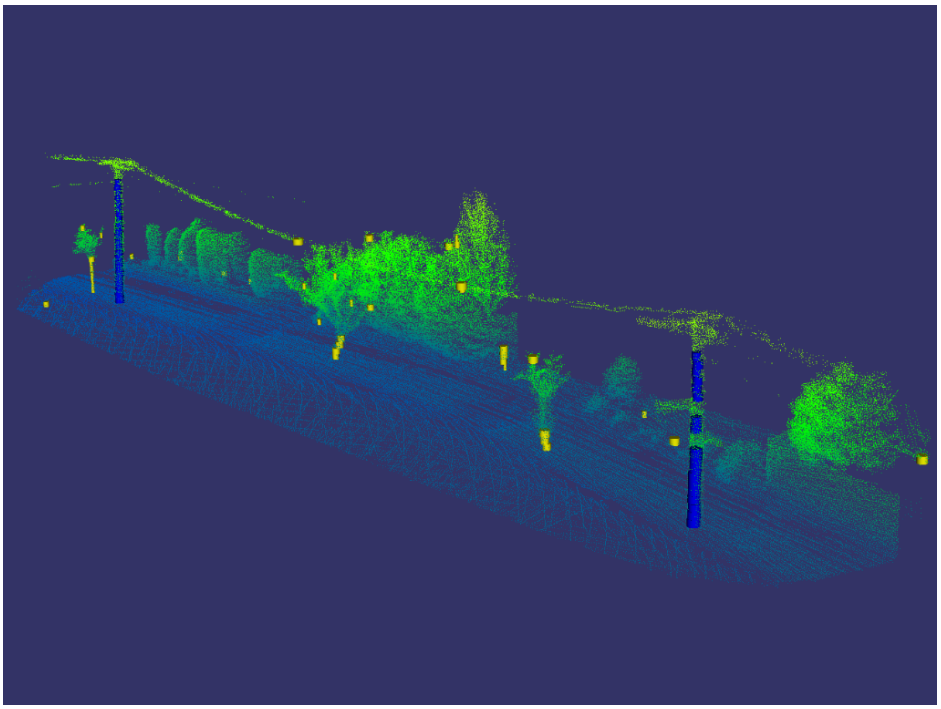


Figure 4.3: Pole slices are rendered as cylinders spanning between the two layers it was create from. The radius is representative of the underlying points. Segments which are successfully stitched into poles are rendered blue while the others remain yellow.

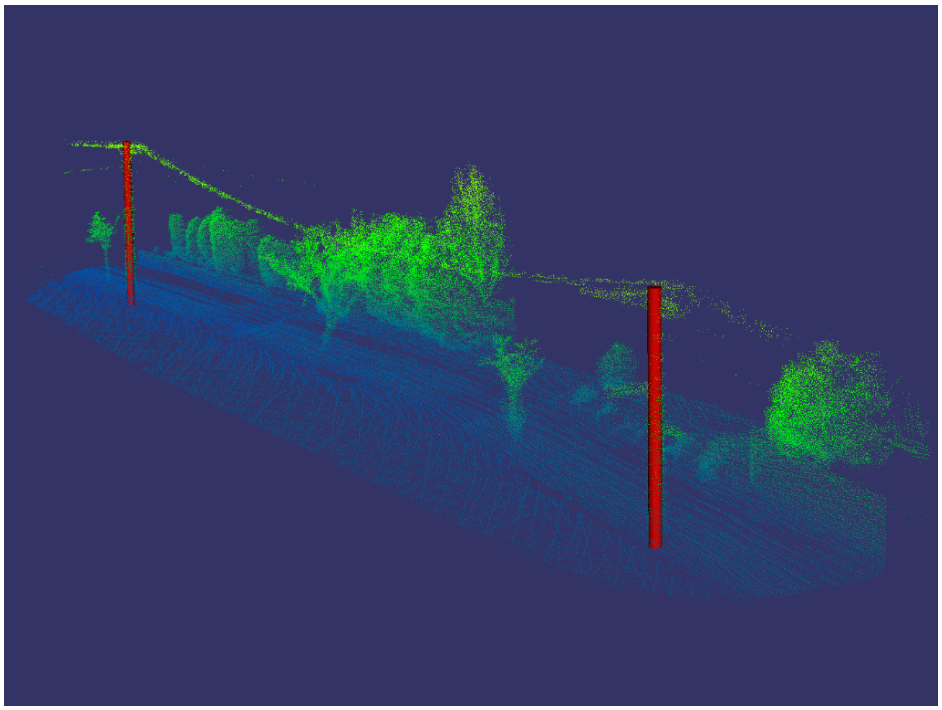


Figure 4.4: Pole segments are rendered as vertical cylinders spanning between the recovered pole's bottom and top and with radius determined from pole slices used to create it.

Pole Segment

The pole segment is used to represent a known pole in the point cloud. It contains multiple additional variables to represent the pole's radius, height, and orientation. It also contains a 2-Dimensional representation of the distribution points making up the pole called a spin image. Each of these variables are discussed in further detail in Pole Recovery chapter. Pole segments are rendered as tall red cylinders as shown in Figure 4.4.

Link Segment

The link segment is used to represent the region between two adjacent poles where there is a high likelihood of conductors being found. The pole link maintains a reference to the two poles it is between and a list of the points in the region of interest. The pole link segment is not normally rendered because it would obscure the conductors within that region, but it is shown

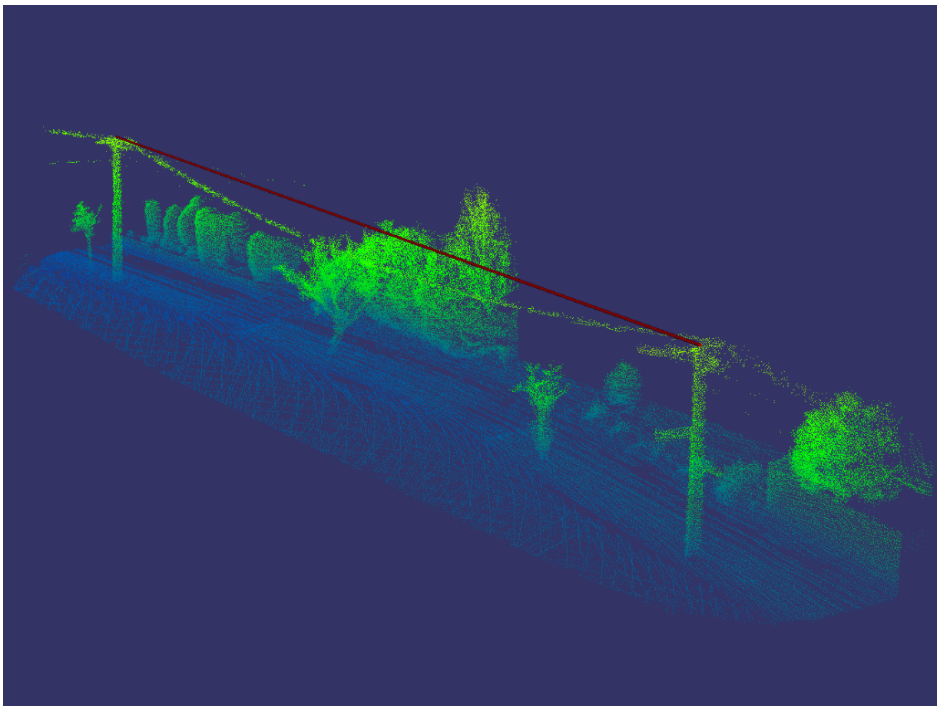


Figure 4.5: The pole link segment is rendered as a line between the two associated pole tops.

in Figure 4.5;

Wire Cluster Segment

The wire cluster segment is used to represent a conductor within the point cloud. It maintains references to the two poles it is suspended between and the points that make it up. It also contains the model parameters used to parametrically describe the conductor. Wire clusters are usually rendered as yellow cylindrical curves, see Figure 4.6.

4.3.2 Classifiers

While segments describe some known information about the point cloud, classifiers describe the method to ascertain said information. Classifiers are invoked by the pipeline in a predefined order and have access to the raw point cloud, along with any previously created segments. All classifiers inherit from a base classifier and are provided with a common interface to the pipeline.

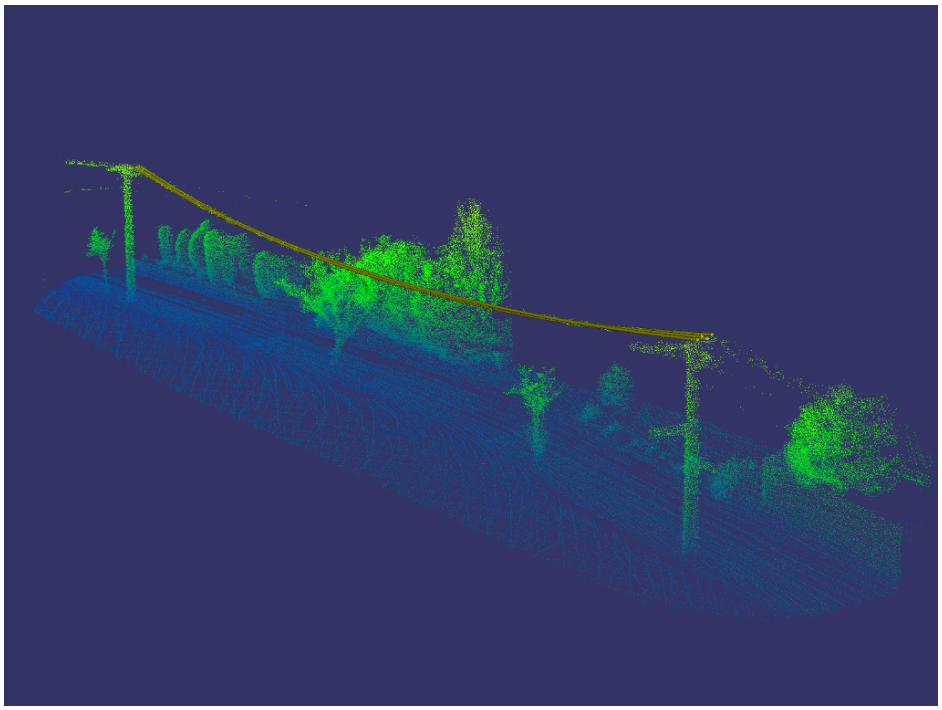


Figure 4.6: Once a parametric model has been fitted to the conductor it is rendered in yellow.

There are two classifiers involved in the process of recovering poles; these are the Pole Slicer, and Pole Stitcher, which are discussed in depth in the Pole Recovery chapter. There are three classifiers involved in the recovery of conductors; these are the Pole Linker, Wire Clusterer, and Wire Fitter, which are discussed further in the Conductor Recovery chapter. There are also two additional classifiers for recovering the terrain and foliage within a point cloud, the Ground Patcher and Bush Detector respectively; these are discussed later in this chapter.

4.3.3 Pipeline

The pipeline provides the framework for classifying point clouds. It maintains the collections of classified segments and classifiers. The classifiers are stored in a data structure referred to as a chain, and invoked sequentially. When a classifier is invoked, the pipeline provides it with references to the point cloud and the collection of segments. By providing a reference to these data

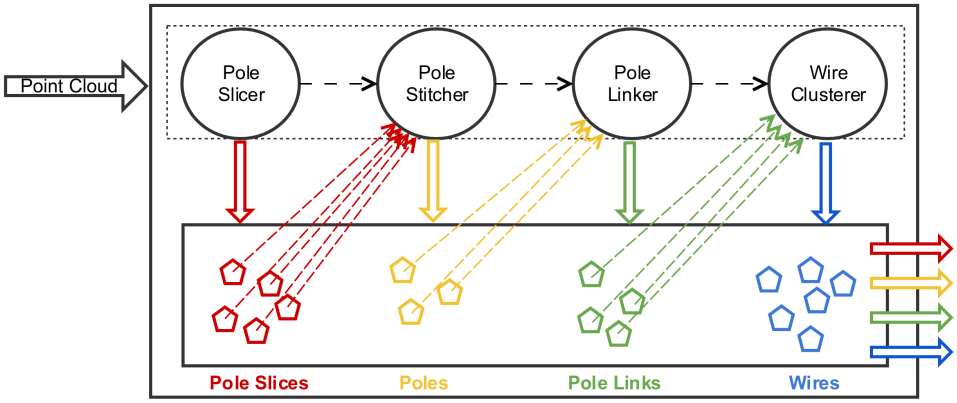


Figure 4.7: This image shows a high-level view of the operations which the pipeline performs. After the point cloud is loaded into the pipeline, each classifier (circle) is executed sequentially based on its position in the chain (dashed box). As each classifier executes, it pushes any segments it created into the pipeline's collection of segments (lower box). Any previously created segments can be used by the classifiers (checkered arrows).

structures, any changes made by a classifier can be seen by any later classifier. Figure 4.7 shows a visualisation of this process.

4.4 Integration

The Classification Pipeline is compiled into a single dynamically linked library. This allows it to be built into both desktop applications and headless services such as would be found hosted in the cloud. In our work, we have only linked this library against our test application.

Our test application is responsible for loading the point cloud into the pipeline and assembling the classifier chain as desired. This test application makes use of OpenSceneGraph, a graphics framework, for rendering. Most of the 3-Dimensional visualisations in this thesis were rendered using this test application.

4.5 Miscellaneous Classifiers

The two classifiers discussed in this section were not developed as part of our primary research goals. Instead, they were necessary to provide contextual information to the pole and conductor focused classifiers.

4.5.1 Ground Patcher Classifier

The ground patcher was developed to provide a digital elevation model (DTM) of the input point cloud for subsequent classifiers. Accurate estimation of the ground elevation is required to estimate pole height and conductor clearance.

Existing Methods

There are two common approaches to recovering a Digital Terrain Model (DTM) from LiDAR collected data, the first looks for the lowest point in a given region, and the second looks for flat regions.

When looking at the lowest point in a region several techniques can be utilised. Aschiff et al. used a simple method of dividing the scan using a horizontal grid with cells of 50 cm along each side, the lowest point in each cell used to build up the DTM [64]. While this approach is easy to compute it is also prone to failing in cells where the ground is not observed; Aschiff et al. partially overcame this by disregarding any point higher than 10 meters above the scanner. Instead of only using the lowest point in a region, Landa et al. used a locally calculated threshold to specify which points will be incorporated into the DTM [65]. The threshold used by Landa et al. is calculated based on the distribution of points within the cell and compared with the global distribution; in their research they tuned the threshold to select points likely to be less than 0.5 meters above ground level. Mongus et al. proposed an approach which recovers the DTM by developing intermediate terrain models of increasing resolution[66]. Like Aschiff, Mongus also uses the lowest point in a given region but applies a series of morphological filters to remove noise, such as points below the terrain caused by reflections, and disjoint regions caused by structures, like houses and bridges. This is

made possible because a lower resolution DTM can be used to inform higher resolution stages of the likely ground location.

DTM recovery methods based around looking for flat areas in the point cloud tend to be used when a much higher resolution of the ground is available, but may not be fully observed. Point clouds with these characteristics are generated when the LiDAR is close to ground level such as on a car, rather than an aircraft which tends to generate more uniformly observed regions. Jaakkola et al. developed a method for exactly this situation [67]. Jaakkola first uses a classifier for identifying roadside kerbs. A triangulated irregular network (TIN) is then used to recover the road surface. This approach is shown to recover a high-quality DTM, but unfortunately, cannot be used to recover regions beyond the road kerb. Golovinskiy et al. used a method based on local plane fitting [45]. Rather than confining the search region like Jaakkola's method, this operation is applied to all cells throughout a horizontal grid. When a near horizontal plane is successfully fitted to a region of points it is incorporated into the DTM. Calberg et al. described a method based around finding the principal components for local groups of points at regular intervals[68]. Calberg's method defines two region types, planar and scatter, which are defined based on the ratio of the region's eigenvalues. A region growing algorithm is then used to build up larger planar regions. Finally large, low, planar regions are selected to become part of the global DTM.

Our method

In our ground classifier, we chose to use a method looking for local planar regions rather than the lowest point. This is due to the non-uniform distribution of points generated by our DAS and the large shadowed regions caused by general ground clutter.

Our method first filters out points with a low likelihood of belonging to the ground. This is done by first finding the principal components of the region around each point. In our implementation, the region of interest contains all points within 0.6 meters of the point of interest. We check that the region is planar by checking ratio between eigenvalues; $eigenA \approx eigenB \gg eigenC$.

We then check that the normal component is within 8° of vertical. This removes all planar surfaces which are not close to horizontal. All points that meet these criteria are then down sampled using a voxel filter with a leaf size of 1 meter. This is the equivalent of taking the centroid of every cell in a 3-Dimensional grid with edge size of 1 meter. This leaves us with enough resolution to represent the terrain and provides us with a uniform point density. Finally, we remove outlier points using the Statistical Outlier Removal (SOR) algorithm. SOR computes the global mean distance between a point and its k nearest neighbours. If any point is significantly further away from its neighbours than the global average it is considered an outlier; this requires two passes through the dataset. This removes all false ground segments which are alone, or in small clusters on top of buildings, and cars, which account for the majority of false positives.

4.5.2 Bush Detector Classifier

The Bush Detector classifier is used to recover the canopy of trees and plants in the scanned point cloud. By recovering the canopy, we can draw comparisons between scans at different dates and measure the rate of encroachment towards the conductors. We are also able to use the classified bushes to search for any minimum clearance violations that may be occurring in the network.

Existing Methods

When parametrising a tree there are two general approaches, either locating the trunk first or the canopy. The trunk of the tree is a good starting point for tree recovery algorithms because it offers a solid surface with little ambiguity to its location. For trees that are maintained, such as those found in pine plantations, it may be sufficient to fit cylindrical models to the trunks using RANSAC, as proposed by Kelbe et al [69]. It is however likely that a more complex model is required to accurately represent the tree's trunk; in these cases a method which processes the trunk as smaller horizontal slices may be used [64, 35]. The slice based method proposed by Bienert et al. searches each slice for trunk cross-sections which should appear as circular structures,

it then searches for vertical stacks of these tree cross-sections. If more than five cross-sections are found stacked upon each other, the rate of taper is calculated. If the taper is within an acceptable range, a conical frustum can be fitted to represent the tree's trunk. These slice based methods can be robust to deformation within a small number of slices and still recover the trunk. We discuss these methods in further detail later in the pole recovery chapter.

When tree recovery is performed from aerial LiDAR other methods are required. This is because the tree's trunk is likely highly shadowed by its canopy. Methods which are based on canopy classification generally exploit the fact that the canopy is partially opaque and will return LiDAR points not only from the canopy surface, but also from its internal structure. Clode et al. proposed a method which utilises a LiDAR with dual return [21]. Clode's method looks for points on the same LiDAR pulse with a large vertical separation; this would be the case when the laser hits part of the canopy and continues on to hit the ground below. When a dual return LiDAR is not available, methods which look at the distribution of points can be used to detect the canopy. Both Carlberg et al. and Lalode et al. use a scatterness measure calculated using the principal components in small regions of points [68, 70]. Regions with a near uniform point distribution will have a high scatterness value and are tagged as likely belonging to the canopy of a tree.

Our method

The method employed by the Bush Detector is very similar to that of the Floor Patcher, but instead of searching for planar regions, it searches for regions of uniform point distribution. We use a scatterness measure to detect regions with a high likelihood of belonging to part of a tree's canopy.

The Bush Detector first down samples the point cloud using a voxel filter with a leaf size of 0.5 meters. For each of the down sampled points, a region of interest is defined which contains all points in the original point cloud within 0.5 meters of the current point. The principal components of distribution of the points within this region of interest are found. The distribution is flagged as a potential Bush Segment if $eigenA >= eigenB >= eigenC > 0.3$,

which is the case for a near uniform point distribution. Finally, a Statistical Outlier Removal filter (SOR) is applied to remove Bush Segments with few neighbours; this is found to remove the majority of false positives.

Chapter V

Proposed Method for Utility Pole Recover

5.1 Introduction

In a vehicle-based LiDAR scan of a utility network, the conductors make up a tiny portion of the points within the scan. Recovering these conductors is made more difficult by their unknown shape, orientation and location within the scan. For these reasons it was decided that first locating the utility poles, and using these to inform the conductor search was an ideal approach. Utility poles are generally near vertical and have heights between 6 and 12 meters. The method discussed here exploits these known characteristics of utility poles to help overcome some of the challenges of working with vehicle-based LiDAR scans.

5.2 Methodology

5.2.1 Slicing

One issue with working with LiDAR data collected from a moving vehicle is the non-uniform density of points within a point cloud; this can be due to occlusion by foreground objects [36] and varying angles of incidence [37]. These non-uniform densities can make small-scale features like surface normals challenging to recover on objects of high curvature like utility poles. To improve the point uniformity, a method of slicing and projecting was used, which is a common approach when extracting vertical pole-like structures [38, 39]. In this implementation the scan is divided into a set of discrete horizontal layers with a thickness of *SliceThickness*, see Figure 5.1. All points within each slice are then projected onto a horizontal plane. By projecting vertically onto the slice plane, the point density can be increased with

minimal information loss and distortion to vertical structures.

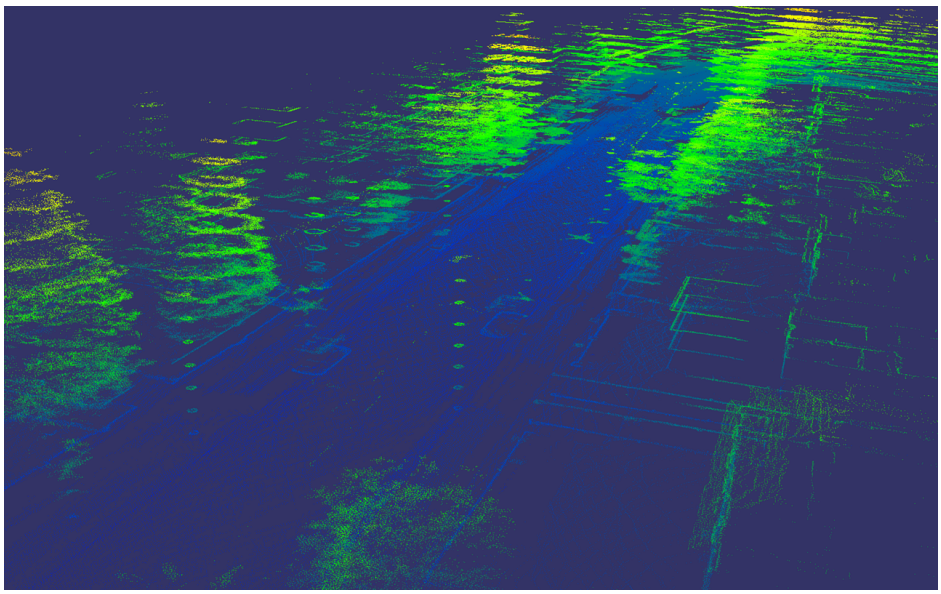


Figure 5.1: Example of a point cloud after horizontal slicing and projection

5.2.2 Pole Segment Recovery

Once all points within the cloud have been projected onto their respective planes, the search for pole segments can be initiated. This is a multi-pass operation which first uses a Euclidian distance based clustering method to identify all point clusters within a slice. Any point within 0.3 meters of an existing cluster is added to it. Otherwise, a new cluster is created. After all of the points have been clustered, any clusters consisting of less than 10 points are disregarded. The second step is to reject clusters that do not reassemble that of a pole cross-section. The segment rejection process first locates the centroid of the cluster. The distance threshold *PoleRadiusThresh* is defined, and the proportion of points beyond this from the centroid is found. If more points lie beyond the *PoleRadiusThresh* than defined by the *PoleInlierPortion* value, the cluster is rejected. This radius based rejection model is similar to the rejection criteria used by Lehtomaki et al. who defined a cylindrical rejection model which is applied in 3-Dimensional space [39]. Figure 5.2 shows an example of accepted pole segments. Any pole segment

that passes these rejection rules then has its radius defined as the mean distance from the centroid for all of the clustered points.

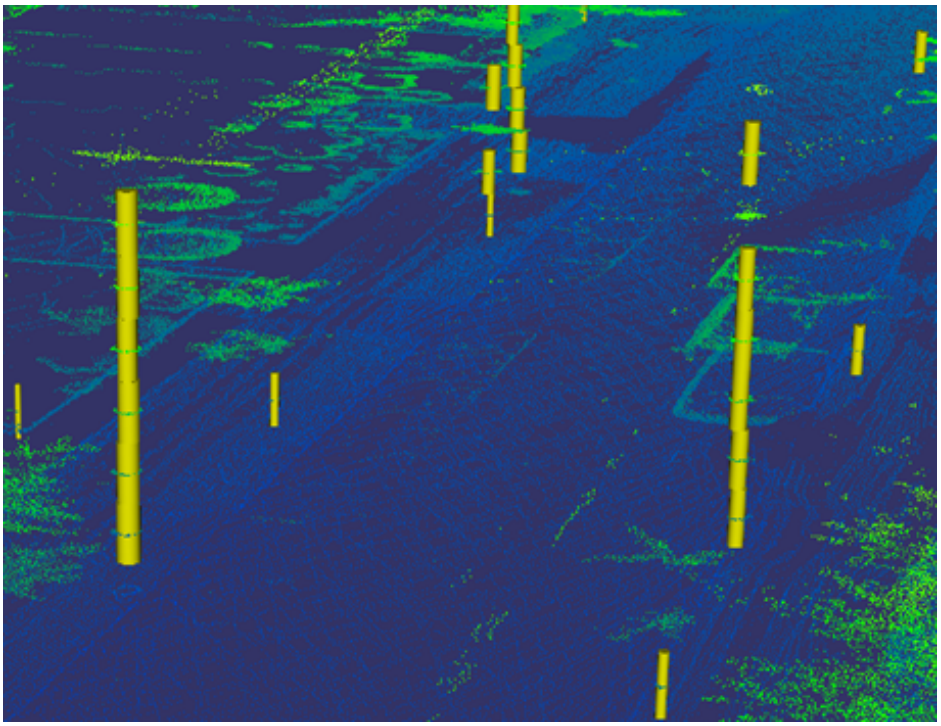


Figure 5.2: An example of accepted pole segments superimposed over the sliced point cloud. Note there are many singular false positives, but the poles are well represented as stacks of pole segments.

5.2.3 Segment Stitching

The pole segment recovery step generates a large number of pole segments which are primitives used to reconstruct complete poles. Many of these pole segments will be false positives, and instead of belonging to a pole, will represent a small area of the point cloud where the local point distribution is similar to a pole-like structure. The segment stitching operation is used to reject a large number of false pole segments and begin the pole recovery. Starting with the lowest most layer, each pole segment is used to seed a pole which is grown up through the layers searching for additional pole segments which are aligned with the existing pole. A pole segment is aligned when

its cross-section overlaps that of the uppermost pole segment currently belonging to the pole. If a new pole segment satisfies this alignment condition, it becomes the new uppermost pole segment of the pole. This method of stitching poles is robust to breaks in pole segment continuity because it does not only search the next layer but all higher layers.

Once all pole segments have been processed into potential poles, the pole rejection conditions are applied. The first condition checks the number of pole segments used to create the pole. Instead of defining the pole coverage directly as a minimum number of pole segments, it is defined as vertical distance *PoleSegCoverage*. For instance, with a *PoleSegCoverage* of 2 meters and a *SliceThickness* of 0.25 meters, a minimum of 8 pole segments would be required for a pole to be preserved. The second rejection criteria looks at the height difference between the bottom-most pole segment and the topmost. Any pole like structure with less vertical coverage than *PoleSegDeltaHeight* will be rejected. It is important to note that this metric does not directly measure the height of the pole as that is not known yet. Instead, this is an indicator of pole height and as such, the threshold is set to a conservative value, approximately half the height of the target poles.

5.2.4 Pole Parametrisation

The first step of pole parametrisation is to find a point that lies along the pole's major axis. This point is found by taking the centroid of the location of each pole segment used to define the pole. The orientation of the pole is found using Principal Component Analysis (PCA). By performing PCA on each of the pole segments' locations the major axis of the pole can be recovered. Using these two data points, pole centroid and major axis, we can fully constrain the orientation of the pole and constrain the location to somewhere along that axis. Finally, the pole radius is defined as the mean radius of each of the pole segments used to define it.

5.2.5 Pole Refinement

After pole parametrisation, it is important to perform some additional refinement to find the bottom and top of the pole accurately. To find these

Table 5.1: Algorithm Configuration

Description	**Variable**	**Nominal Value**
Slice Layer Thickness	*SiceThickness*	0.25 meters
Pole Radius Threshold	*PoleRadiusThresh*	0.30 meters
Pole Inlier Proportion	*PoleInlierPortion*	10%
Pole Required Coverage	*PoleSegCoverage*	2 meters
Pole Required Height	*PoleSegDeltaHeight*	4 meters
Pole Refinement Radius	*PoleRefineRadius*	0.5 meters

extremes, each point within *PoleRefineRadius* from the pole's major axis is projected onto the major axis. If that projected point is more extreme than any before it, it will become the new top or bottom of the pole.

This completes the pole recovery process as each pole's location, orientation, radius, and height has been found.

5.3 Results

The proposed method's primary goal is to accurately find the top of utility poles to seed the search for inter-pole conductors. Two metrics were used to assess performance. The first is a measure of the precision and recall of the pole detector, and the second is a comparison between the computed pole height and that found by human inspection of the point cloud. For all testing, the algorithm was configured with the default values as seen in Table 5.1.

5.3.1 Pole Recovery

The locations of all 37 utility poles in the Gilberthorpes dataset were successfully recovered, see Figure 5.3, along with the two partially observed lamp posts off the main street. All 13 utility poles in the more cluttered Hounslow dataset were successfully recovered. There were no false positives in either dataset.

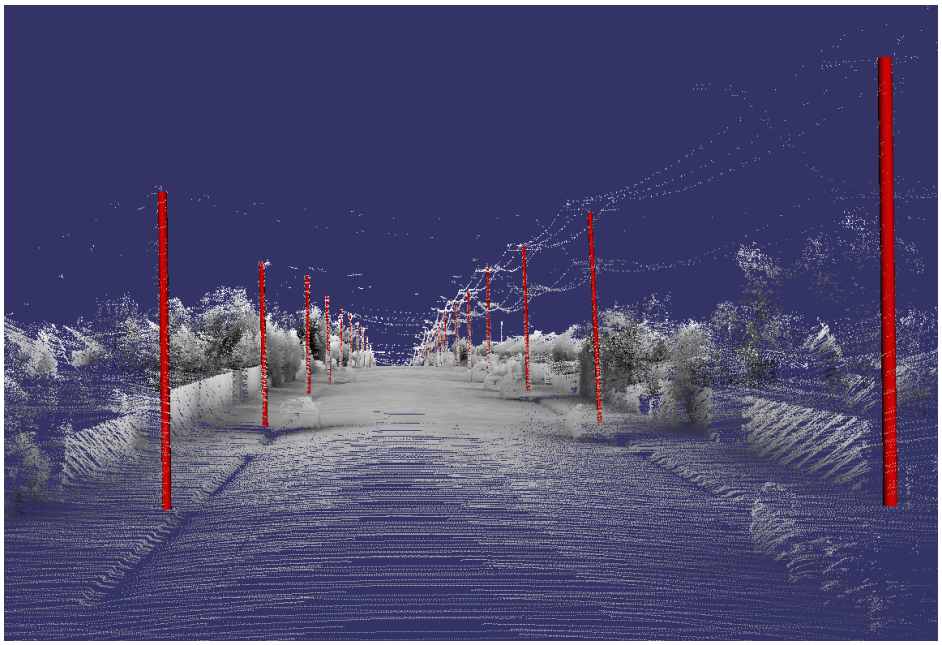


Figure 5.3: Example of pole recovery (Gilberthorpes).

5.3.2 Pole Height

For the 37 poles in the Gilberthorpes dataset, the height values computed by the proposed algorithm were compared with the measured heights. Overall with the 37 poles, the mean difference between the computed height and measured values was -8mm with a variance of 25mm.

Of the 37 poles, there was one significant failure to detect the height of the pole correctly, see Figure 5.4. This was where the lower portion of the pole was entirely shadowed by a vehicle parked in front. In this case, the pole was computed to be 0.78 meters shorter than the human measurement. There are two points to note for this measurement. Firstly the human measured height is of degraded reliability due to the difficulty in identifying the bottom of the pole. Secondly, a visual inspection shows that the top of the pole was recovered with a similar accuracy to that of any other pole in the dataset.

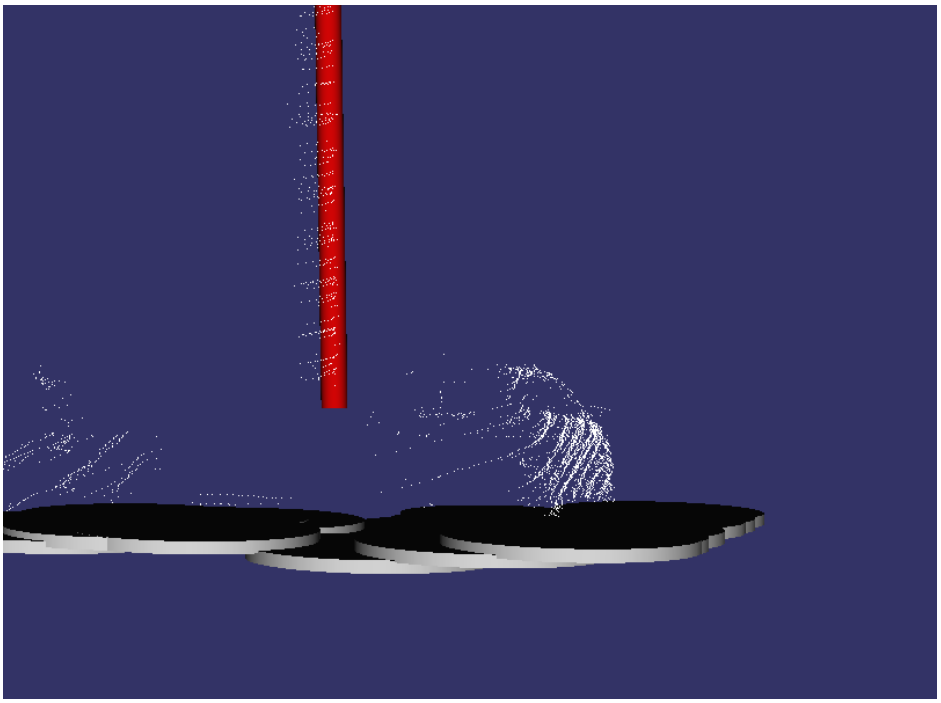


Figure 5.4: Failure to detect the lower portion of the utility pole due to complete shadowing from a parked vehicle.

5.4 Discussion

The described method performed well on the two datasets with the recovery of all 50 poles. This performance compares well to other modern pole recovery methods such as that proposed by Cabo which achieved a recall of 94% over 57 poles. Cabo's method was also designed to recover trees and other pole-like street clutter, it is to be expected that the utility pole recovery performance is reduced when targeting multiple object types.

While a high recall rate is essential, it is also necessary that the pole tops be accurately located for seeding the conductor search. The metric used to judge a pole top's accuracy was pole height, as it is an easy and reliable measurement to make. This metric indicated good performance of pole top localisation with an average height error of 8mm and a standard deviation of 25mm. It is worth noting that the actual pole top accuracy is likely higher because the accuracy of the height is dependent on the accurate recovery of both the top and bottom of the pole. Locating the bottom of the pole can

be error-prone due to ground clutter shading this area; an example of this can be seen in figure 5.4.

While the method discussed can recall all poles in the dataset and accurately locate the pole tops, in situations where the pole is not well observed from all angles it can be pulled off centre towards the direction of observation. This is because the pole location is found using the points' centroid to find the centre of the pole segments. A more intelligent method of finding the centre of pole cross-sections such as a geometric centre could reduce this effect. While this tendency for poles to be slightly off centre is a limitation of this method, the effect is not significant enough to be detrimental to the recovery of conductors.

This pole recovery method makes assumptions that the pole is upright, and a significant portion of not necessarily connected regions of the pole will be scanned by the LiDAR. Using these assumptions, and variables to describe the height and radius of the target poles, the location, orientation, and size of poles within a LiDAR scan can be reliably recovered.

Chapter VI

Proposed Method for Conductor Recovery

6.1 Introduction

As discussed in the Foundational Work chapter, the automation of conductor recovery in urban environments has not been well explored. The work that has been done as approached the problem by trying to isolate the conductor points through a series of filters. The method presented in this chapter attempts to make use of the non-conductor points within scan rather than disregarding them. This approach to conductor recovery is designed to overcome some of the challenges associated with operating with data collected from vehicle-mounted LiDAR. Also outlined are the the method's short comings and the false assumptions that were made; both of which will be addressed in the following chapters.

6.2 Methodology

6.2.1 Conductor Search Space

One unfortunate side effect of LiDAR data collected from a ground vehicle is a significant amount of strand-like sequences of points occur in regions that are only partially observed (See figure 6.1); particularly near the periphery of the scan where there can be a lot occlusion from other objects. These strand-like artefacts are not commonly found in aerial LiDAR scans because the vantage point provides less opportunity for objects to shade each other. To reduce the likelihood of the many strand-like sequences of points interfering with the conductor recovery process, we define a sub-section of the point cloud between two utility poles as the conductor search region. This conductor search region is generally well observed as it is near the path travelled by

the ground vehicle; as such it does not contain many strand-like sequences of points.

Before the conductor search space can be defined, the set of all utility poles is searched for pairs of poles which could potentially share a conductor span; this is based on their distance apart. In the following work an inter-pole distance of 50 meters was used; approximately 20% greater than any observed spans in our datasets. This value is generally known ahead of time, based on the type of utility poles being scanned.

Once a pair of poles which are close enough to be connected with conductors is found, the conductor search space can be defined. The conductor search space is a cuboid shape with the two poles being located in the centre of opposite faces; thus defining the length of the region. The width of the region is three meters, and it extends vertically to include all points; effectively infinity.

To reduce the degrees of freedom for the remaining conductor recovery steps, the points within the conductor search space are transformed into a local coordinate system. This transformation between the global and local coordinate systems is comprised of a rotation (See Equation 6.1) and translation (See Equation 6.2) component. Both of these are calculated using the pole top locations for the two poles at either end of the span ($pATop$ and $pBTop$). The resulting conductor search space transformation is shown in Equation 6.3. For the remainder of this chapter we will refer to the three axes of this coordinate space as spanwise (horizontal direction of the conductors), cross-spanwise (horizontal direction perpendicular to conductor direction), and vertical.

$$rot = \text{atan2}(pBTop.Y - pATop.Y, pBTop.X - pATop.X) \tag{6.1}$$

$$trans = \begin{pmatrix} pATop.X & pATop.Y & \frac{pATop.Z + pBTop.Z}{2} \end{pmatrix} \tag{6.2}$$

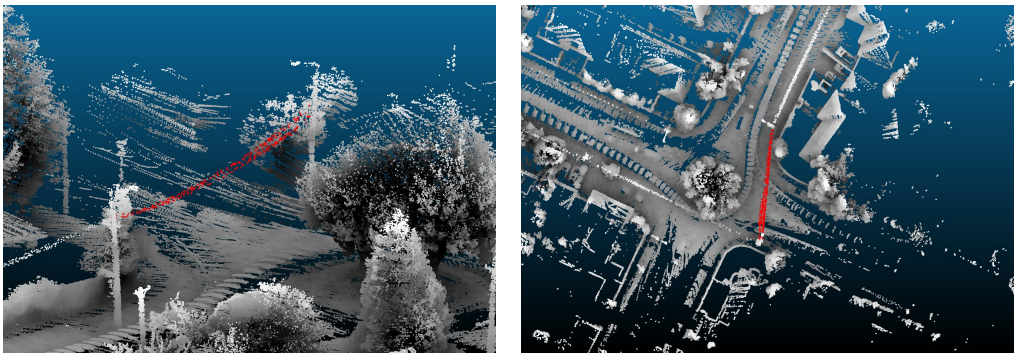


Figure 6.1: These two images show the many strand-like sequences of points contained within a scan; including one conductor span highlighted in red.

$$transformation = \begin{bmatrix} \cos(rot) & -\sin(rot) & 0 & trans.X \\ \sin(rot) & \cos(rot) & 0 & trans.Y \\ 0 & 0 & 1 & trans.Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.3}$$

There are two other significant benefits of using this conductor search space. Firstly, the number of points to be searched is drastically reduced from the entire point cloud to generally less than one hundred thousand. Also, the two endpoints of the conductor search space are defined by the utility poles, any conductors within the search space can be assumed to be close to parallel to the semi-major axis of the search space.

6.2.2 Clustering Plane

Before conductor models can be fitted to the points within the conductor search space, the points need to be clustered into individual conductors. Clustering points into conductors can be challenging due to the low density of points representing them.

The density of points is so low that regularly the nearest neighbour of a point will be on an adjacent conductor; instead of further along the same conductor. This is problematic for any algorithms which cluster based on the spatial separation of points. Higher level features can be derived from the conductor points to inform the clustering algorithm; such as using the

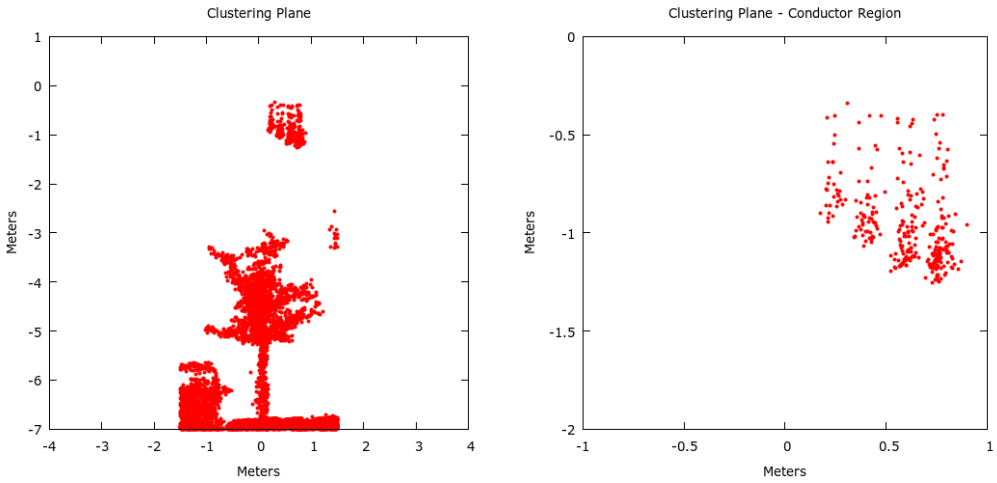


Figure 6.2: Example of the points within a conductor span after projection onto the clustering plane. Left is the complete clustering plane, right is a closer look at just the conductors

eigenvectors of local regions to traverse along the conductor spans [28]. However, the proximity of conductors and low point densities in our scans would require that the windows used to calculate these features be so large as to include multiple conductors; thereby having less value in separating them.

Despite the low point density hindering the effectiveness of spatially based clustering, it makes sense to attempt to use these methods because we know they are spatially separable. To improve the density of the conductor points, the conductor search space is collapsed along the span-wise direction. This transformation converts the problem from clustering 3-Dimensions to clustering on a 2-Dimensional plane perpendicular to the span-wise direction (See Figure 6.2); referred to as the clustering plane. During the collapse of the conductor search space all points closer than 2.5 meters to a pole are ignored. This is to avoid cross-arms and other pole decorations such as street lights being included on the clustering plane. This distance of 2.5 meters is referred to as the "pole ignore radius".

When projected onto the clustering plane, the points belonging to the conductors form vertical stripes. This structure is due to the shape of the conductors; with the lowest most points of the span at the bottom of the stripes. These stripes will always be vertical because the conductors are

hanging freely.

6.2.3 Cross Span-wise Density-based Clustering

The Cross Span-wise Density-based Clustering (CSDC) algorithm was developed to exploit the vertical structures present on the clustering plane. It was a simple method to act in place of more complex clustering methods, while the rest of the conductor recovery pipeline was developed. While CSDC was developed as a placeholder, it is a surprisingly effective clustering method, and elements of its design are present in the following described clustering algorithms.

The CSDC algorithm performs clustering by attempting to group points based on their local density along the cross span-wise axis. This cross span-wise density can be seen in Figures 6.3 and 6.4. Early implementations of this method used a histogram to measure the point density and a hard threshold to segment regions of the histogram into clusters. There are two significant limitations to this approach. Firstly, there is a trade off when selecting the resolution of the histogram; buckets too large may not be able to distinguish different conductors, while too small and local noise can become a problem. Secondly, the hard threshold requires that the least dense cluster must be more dense than the most dense region between clusters; an assumption that is not always true as seen by the PDF in Figure 6.3.

To overcome the limitations of the initial histogram based CSDC the histogram was replaced with a probability density function (PDF). The PDF is found using the Kernel Density Estimation algorithm with a Gaussian kernel. With the new underlying continuous density model, the threshold function was replaced with the mean-shift algorithm [71]. The mean-shift algorithm uses only local changes in density and thus can find clusters with densities lower than that of other inter-cluster regions. A threshold function is still used to cull clusters of very low-density which is trivial using the PDF.

By using the mean-shift implementation of the CSDC, single layers of conductors can be reliably clustered. However, this algorithm is based on the assumption that clusters are separable using the point density along only the cross span-wise direction. This assumption can be broken when spans

have multiple levels of conductors. For these spans, if a conductor on an upper level is directly above a conductor on a lower level, the conductors will not be separable using only the cross span-wise density; an example of this can be seen in Figure 6.4.

One other significant limitation is that the CSDC cannot cull out points belonging to non-conductor objects within the conductor search space; such as the ground and ground clutter. The CSDC uses a hard threshold to remove points too far below a vector linking the two pole tops. For the multi-level conductor span shown in Figure 6.4 a threshold of five meters was used to include both layers. However, in the single layer example, a threshold of five meters will include a significant amount of ground clutter that will pollute the density function; a threshold of two meters is used instead. This value needs to be dynamically found, but the CSDC currently has no way of doing this and the value is manually provided. Because the CSDC inherently cannot separate vertically aligned conductors, we decided not to attempt to solve this dynamic threshold problem. The two clustering algorithms discussed in the subsequent chapters have methods of overcoming these limitations.

6.2.4 Conductor Model Fitting

The final step is to fit a model to the clustered points. The conductor model is comprised of two components, the cross span-wise location, and the sag model. By splitting the model into these two components, the sag model can operate to a 2-D plane and only describes the sag for a given distance along the span. While the cross span-wise component describes the 3rd dimension; where the conductor is located along the cross-arms.

The conductor sag model can be selected from one of two models. Ideally, the catenary curve should be used; which describes a cable hanging under its weight. Alternatively, a parabola can be used; which describes a cable hanging with a load distributed along its length, for instance, a suspension bridge. While the catenary is the physically correct model to use, there are two reasons why a parabola may be used in its place. The catenary is a transcendental equation which can be difficult to fit using a least-squares method and can become unstable. Secondly, under the conditions conductors

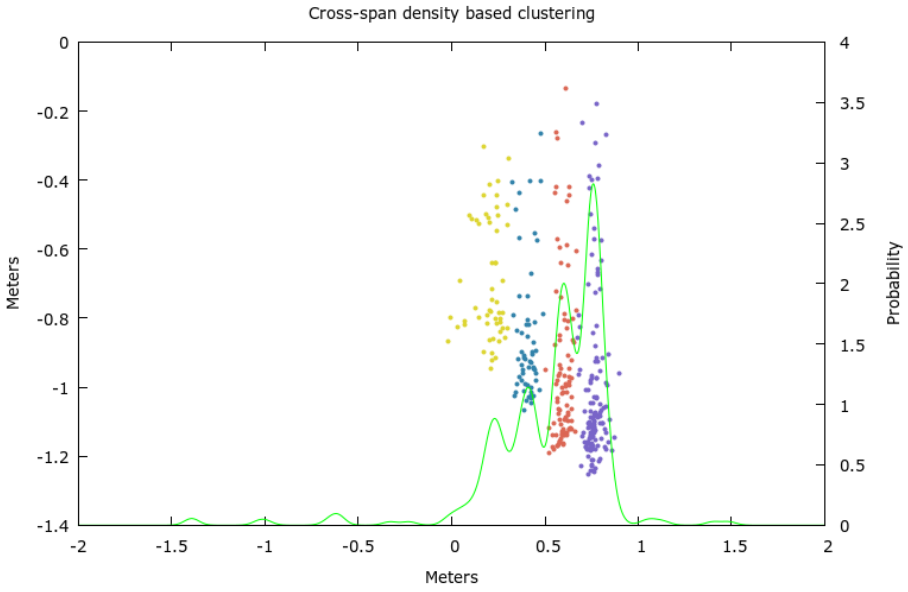


Figure 6.3: CSDC method on a single level of conductors. The cross span-wise PDF shown in green is estimated using a Gaussian kernel with a bandwidth of 4cm.

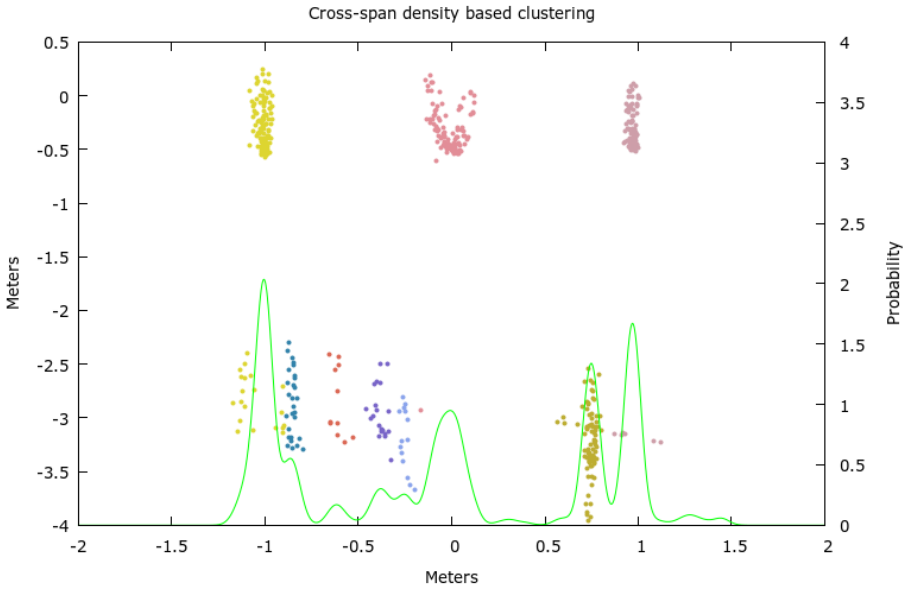


Figure 6.4: CSDC method on two levels of conductors. Note the conductor in the lower left has been clustered with the conductor above.

are normally found, relatively high tension with small amounts of sag, there is little difference between the two models.

In this work, both the parabola and catenary models were used. The parabola (Equation 6.4) can be easily and reliably fitted to the conductor points using the Levenberg-Marquardt algorithm [72]. The catenary (Equation 6.5) can also be reliably fitted assuming good initialisation parameters; most sensitive of which is the scaling factor a. Both models share the values h and k which describe the root of the model, while the scaling factor a differs between them.

Neither of these models accounts for forces other than gravity influencing the shape of the conductors. Wind provides the most significant potential force that could act on the conductors and could cause the conductors to swing like a pendulum. These forces are ignored because it is not possible to scan the conductors with LiDAR in windy conditions because of the long acquisition time.

$$y = a * (x + h)^2 + k \tag{6.4}$$

$$y = a * \cosh(\frac{x + h}{a} - 1) + k \tag{6.5}$$

Once a model has been fitted to the conductor points, it is checked for validity using two rules. The first check is that the shape of the conductor describes a conductor hanging under gravity and not floating upwards. The second check is that the ends of the conductor are at a sensible height; not more than 0.5 meters above the pole top.

6.3 Results

6.3.1 CSDC Performance

Hounslow dataset

The Hounslow dataset is a stretch of road 400 meters long with a significant amount of ground clutter and foliage near the power lines. Along the west side of the road, there are six conductor spans with four conductors and

	Conductors	Communications	Errors
West	24/24	3	2
East	3/3	0	1
Crossing	4/4	0	23

Table 6.1: CSDC Hounslow Results

one communications line. Along the east side, there are five poles and three conductor spans. There are also four spans bridging the two sides. The four conductors running along the west side are the primary interest as they provide power for the entire street. For this dataset, the CSDC used a search window extending two meters down from the pole tops.

Table 6.1 shows the CSDC performance on the Hounslow dataset. On the west side, there were two clustering errors. These were caused by a cross-arm and a tangential conductor entering the conductor search region. There was also an additional cluster on the east side caused by a tree within the conductor search space. The spans crossing the road contained significantly more clustering errors; 14 were tangential conductors and one was from a cross arm. However, the span with the largest number of errors did not contain any conductors but had a tree between the two poles causing eight false clusters.

Gilberthorpes dataset

The Gilberthorpes dataset used is a section of road 150 meters long containing three primary conductor spans, each with two levels of conductors, three on top and four below. There are also three conductor spans crossing the road. For this dataset, the CSDC used a search window extending four meters down from the pole tops. This allows for both layers of the primary conductor spans to be projected onto the clustering plane.

Table 6.2 shows the results of the CSDC on the Gilberthorpes dataset. There were three errors along the primary spans, one of which was within each span. For two of the spans the upper centre conductor was partially split and the other span had the upper centre conductor clustered into one of the lower level's conductors; See Figure 6.4. For the secondary spans,

	Conductors	Errors
Primary	19/20	3
Secondary	3/3	1

Table 6.2: CSDC Gilberthorpes 4m Results

	Conductors	Errors
Primary	9/9	1
Secondary	0/0	1

Table 6.3: CSDC Gilberthorpes 2m Results

there was one false cluster where a street lamp attached to a power pole was hanging within the conductor search space.

Gilberthorpes top layer dataset

This is the same Gilberthorpes dataset as above, but with a search window extending only two meters down from the pole tops. This results in only the topmost layer of conductors being projected onto the clustering plane.

Table 6.3 shows the performance of the CSDC on the Gilberthorpes dataset using a conductor search space extending two meters down from the pole tops. The was one error on the primary conductor spans where the centre conductor on the upper level was split into two clusters and one error on the secondary spans where a street light was within the conductor search space.

6.3.2 Conductor model fitting performance

To verify the earlier claim that a parabola can be used to approximate a conductor under standard conditions, a span containing four conductors was modelled using both parabolas and catenaries. Table 6.4 shows the comparison between the two models. The variables H and K show the fitter root locations; which have less than one millimetre difference between the two models. Importantly the cost function, which was Mean Square Error, is identical to three significant places between the two models. The high

	Parabola			Catenary		
	H	K	Cost	H	K	Cost
Conductor 1	15.7784	-0.8902	0.013723	15.7781	-0.8902	0.013727
Conductor 2	15.9233	-1.0277	0.015553	15.9234	-1.0276	0.015550
Conductor 3	16.0505	-1.1476	0.036869	16.0507	-1.1475	0.036890
Conductor 4	16.0724	-1.2218	0.080967	16.0722	-1.2217	0.080965

Table 6.4: Comparison between parabola and catenary models. Variables H and K describe the span-wise and vertical location of the roots. Cost is the mean squared error of the fitted model.

	Accepted	Rejected
Conductor Cluster	**31**	0
Non-Conductor Cluster	11	**17**

Table 6.5: Conductor rejection for Hounslow dataset

precision of these numbers is given to compare the two models and not representative of the actual accuracy of the models.

As expected there is no visible difference discernible in Figure 6.5. It is only if we significantly extend the models beyond the length of the span the two models begin to diverge; See Figure 6.6.

6.3.3 Non-conductor cluster rejection performance

Hounslow dataset

Of the 58 clusters found by the CSDC, eleven were incorrectly accepted during the conductor fitting process; see Table 6.5. Ten of the eleven false conductors were on a single candidate span which did not contain any conductors but instead contained a tree. The CSDC created ten conductors from this tree, and all of them passed the rejection test. The other false positive was caused by a tangential conductor falling withing the conductor search space. All eleven false positives can be seen in Figure 6.7.

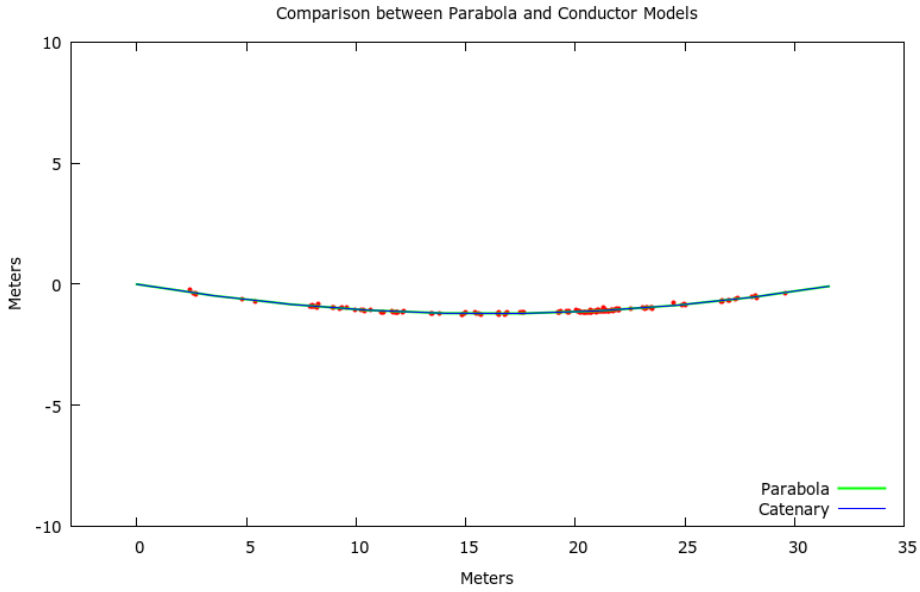


Figure 6.5: Example of parabola and catenary models. The parabola is draw with a thicker line so it can be seen behind the catenary.

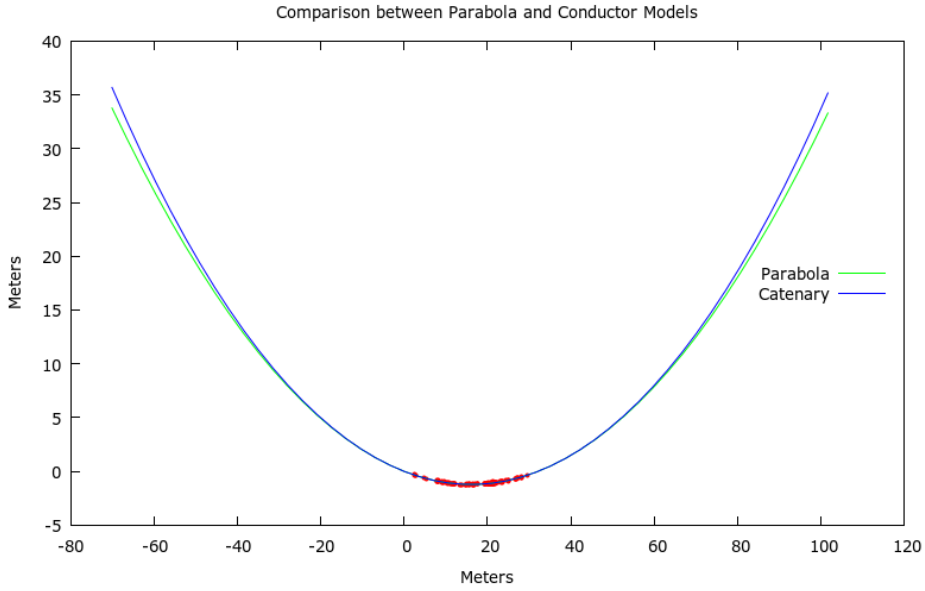


Figure 6.6: The parabola and catenary can only be shown to diverge if the models are significantly extrapolated.

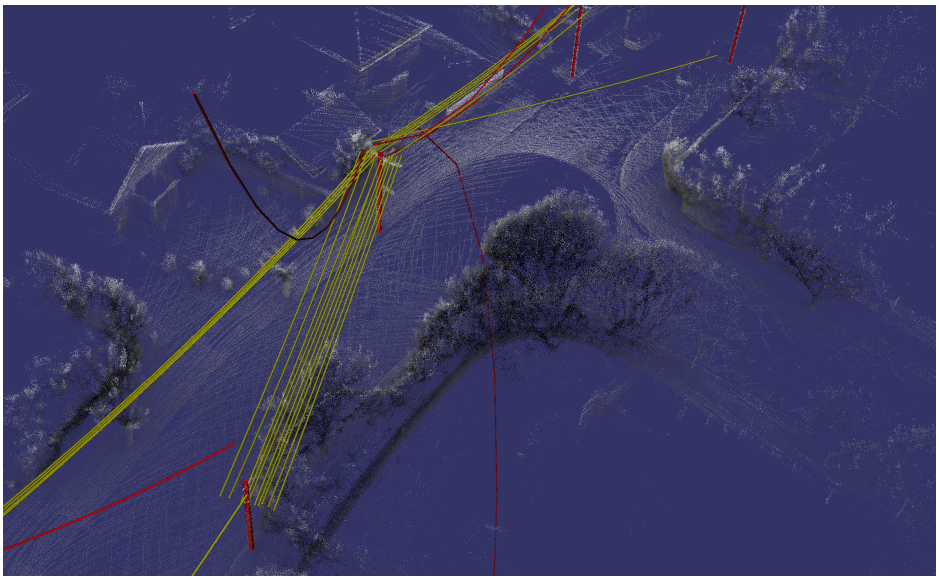


Figure 6.7: Fitted conductors from the Hounslow dataset coloured based on the rejection rules; yellow conductors pass while red are rejected.

	Accepted	Rejected
Conductor Cluster	**8**	2
Non-Conductor Cluster	0	**1**

Table 6.6: Conductor rejection for top layer in Gilberthorpes dataset

Gilberthorpes top layer dataset

The CSDC identified eleven clusters for conductor fitting. Of these eleven clusters, eight were correctly identified as conductors, and one non-conductor rejected; See Table 6.6. Two conductor clusters were falsely rejected because the endpoints of the fitted conductors were above the threshold of 0.5 meters. This poor fitting can be attributed to the low-quality clusters generated by the CSDC; one of which was split into two clusters and the other only clustered a small portion of the span. Regardless they were conductor points that were rejected during the fitting process and thus are marked as errors.

	Accepted	Rejected
Conductor Cluster	**12**	0
Non-Conductor Cluster	4	**5**

Table 6.7: Conductor rejection for both layers in Gilberthorpes dataset

Gilberthorpes both layers dataset

The CSDC struggles to split conductors on the different levels into separate conductors if there is little cross span-wise separation between them. Many clusters contain points from multiple levels of conductors which skews the fitting process. Ideally, the rejection rules should reject any conductors where the fitted model tries to describe more than one conductor. All four of the accepted non-conductor clusters in Table 6.7 are cases where the model was fitted to multiple levels. The rejection rules were otherwise able to distinguish all conductor and non-conductor clusters.

6.4 Discussion

In this chapter, we have presented a method of recovering a model of overhead conductors from vehicle-based LiDAR. This method can overcome challenges like occlusion and variable point density that many earlier attempts were not subject to due to the use of airborne LiDAR or significantly more expensive equipment. Through the use of a clustering plane the inconsistent point density along a conductor span can be compensated for; where conductor traversal methods would otherwise fail.

The Cross Span-wise Density-based Clustering algorithm (CSDC) is shown to be an effective method of clustering single layers of conductors and also able to cluster multi-layer spans with an average of one error per span; caused by conductors being inseparable along the cross span-wise axis. The most significant failure by the CSDC was between two poles containing a tree where conductors could otherwise be found. In this case, eight false clusters were produced.

Many previous works have stated the catenary as being the correct model for a conductor, before switching to a parabola based model with no further

discussion. In Section 6.3.2 it is shown that under the conditions which conductors are found there is, in fact, no practical difference between the two models.

Finally after fitting models to the conductors, a rejection check is applied to verify the conductors are an acceptable shape. This final pass has an accuracy of 81%. A collective false negative rate of 4% and false positive rate of 40% was observed across all datasets. Showing that this rejection system is biased towards preserving conductors; a desired behavouir from such a simplistic final check.

There are of course improvements which could be made to this system, some of which are implemented in the two later chapters. Most notably is an improved method of clustering which accounts for not only the cross span-wise location of the conductors but also the vertical separation between them. Such an improvement should reduce the number of clustering errors in multi-level conductor spans. The most significant single failure of this system was false conductors that were recovered from the search space between two poles which contained a tree. A method of determining if a cluster is uniformly distributed along the entire span instead of a single large cloud, could reduce the occurrence of such errors.

Chapter VII

Proposed Method for Conductor Clustering using Sag Compensation

7.1 Introduction

In the previous chapter, the Cross Span-wise Density-based Clustering (CSDC) method of clustering conductors was presented. While this method showed promising results, under particular conductor configurations the CSDC would tend to produce errors; namely when two conductors were aligned vertically. The Conductor Clustering using Sag Compensation (CCSC) method discussed in this chapter is an evolution of the CSDC designed to be able to robustly cluster conductors, even when they are closely aligned vertically.

The CSDC used a clustering plane when clustering the conductors. When the conductors are projected onto this clustering plane they produce vertical structures due to their sagging shape. These vertical structures can be tightly packed, making 2-Dimensional Euclidean distance based clustering difficult. To overcome this, the CSDC disregarded the vertical positions of the projected points and clustered based on the cross span-wise density. In contrast, the CCSC method discussed in this chapter attempts to maintain the vertical information of these points, but remove their sag component. By maintaining the vertical information conductors vertically aligned should still be separable, while the removal of the conductor sag should increase the density gradient between neighbouring conductors on the same level within the span.

7.2 Methodology

The Conductor Clustering using Sag Compensation method, like the CSDC, operates within a cuboid region which stretches between two poles. This conductor search space extends vertically to infinity and has a cross span-wise dimension of three meters.

7.2.1 Sag Compensation

The sag compensation step is a method of pre-processing the points before projection onto the clustering plane to remove the sag from the conductors and simplify the clustering task. In order to accomplish this sag removal, a model of the average conductor sag needs to be estimated.

To estimate the average sag of all of the conductors the conductor search space is divided into subregions along the span-wise direction (See Figure 7.1; referred to as windows. Within each of these windows, the centre of mass of the conductor points is found. A conductor model is then fitted to these resultant centre of mass points to estimate the span sag.

For this work a window size of 2.5 meters along the span-wise direction was used. It is important that the window size allows for enough points to be included to ensure a stable centre of mass is found. While also small enough to ensure many centres of masses are found for robust fitting of a sag model. With a mean span length of 40 meters, a window size of 2.5 meters gives 16 centres of masses. In practice, this value is less because points closer than 2.5 meters to are pole a disregarded as there are non-conductor points within this region.

Before calculating the windows' centre of mass, the windows need to be further subdivided into panes. This is because the conductor search region includes all points between the poles, i.e ground and clutter. Thus the windows also includes these points. To remove non-conductor points, we first assume that the conductors will be the highest points within a given window. The highest point within the window is found and this will define the uppermost boundary of the new pane. The bottom boundary is then defined as being 0.25 meters below the uppermost boundary. The pane height of 0.25 meters was chosen because it is greater than a conductor would be expected

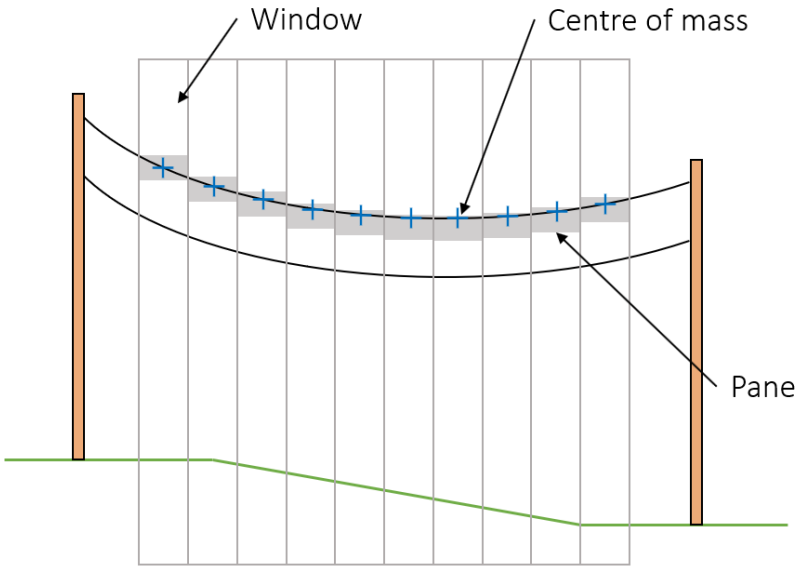


Figure 7.1: Span sag estimation process calculates the centre of mass (blue) for the top most points within each window.

to drop within a single window, but also smaller than the expected clearance between a conductor and an object below.

Using a pane height of 0.25 meters results in only the topmost layer of conductors being used for estimating the span sag. While ideally the sag estimation model would be based on all of the conductors, there are challenges to including multiple layers of conductors within the sag estimation. The primary issue is that within a given window, the ratio of points between different layers of conductors can change significantly; this instability is due to the low number of conductor points within a window. This changing ratio has the effect of pulling the centre of mass towards the conductor level with more points; See Figure 7.2. The second issue with increasing the pane height to include multiple conductor layers, is the number of conductor layers is not known. If the pane height is too large, it may start to include ground clutter, while too small and the lowest conductors may be missed. To avoid these issues, only the uppermost conductors are used for sag estimation.

A filter is applied to remove any centre of mass points lying along the

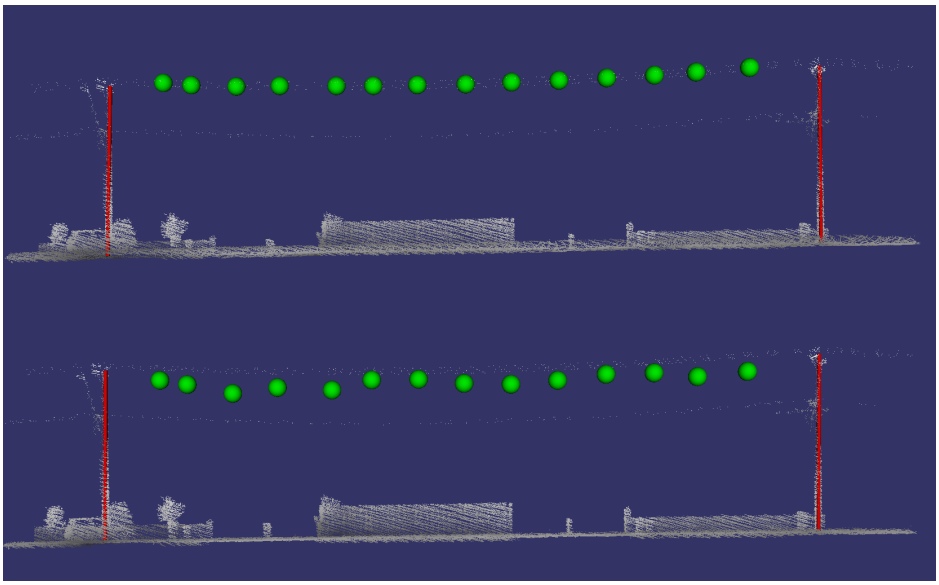


Figure 7.2: Comparison of windows' centre of mass calculation using upper most conductors (top) and all conductor layers (bottom).

ground. This can be the case when there are no conductors between two poles, or the conductors are only partially observed; See Figure 7.3. This check is primarily designed to aid in the recovery of conductors spanning across the road, as these are typically not as well represented as conductors running parallel. These partially scanned conductors can lead to some windows having no conductor points, and thus the centre of mass falls to the next highest object within the window; for spans crossing the road, this is usually the road surface. This rejection check is implemented by measuring the height of each window centre of mass above the vector linking the two pole bottoms. If the height of the centre of mass is less than two meters above the vector, it is not considered during the rest of the span processing. The height of two meters was chosen because it is high enough to include most undulations in the ground while still staying well clear of any conductors.

The 2-Dimensional sag model is then fitted through the windows' centre of masses. The sag model is only 2-Dimensional because it only needs to describe the estimated sag for a given distance along the span. This sag model is in reality just a conductor that is fitted to the window centre of masses rather than clustered conductor points. Using this sag model we

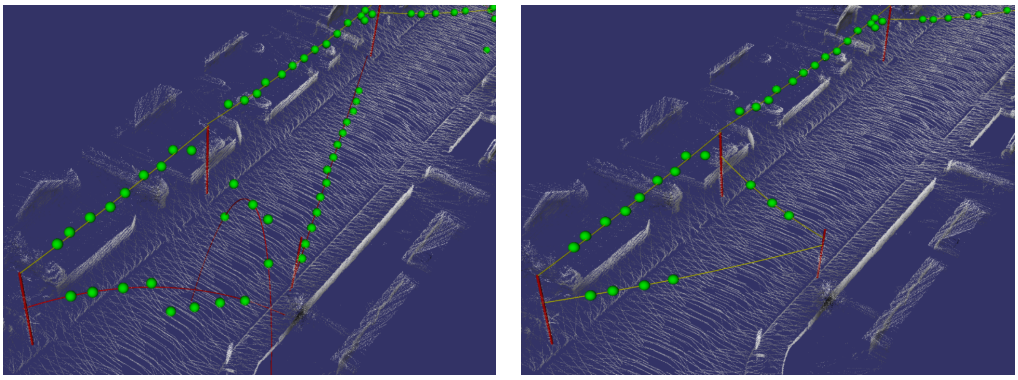


Figure 7.3: Demonstration of no ground centre of mass rejection (left) and with rejection (right). Note the two spans that are crossing the road in the lower portion of the images are pulled down by the ground centre of masses. Also, note this centre of mass rejection inherently discards spans crossing the road with no conductors; An example is shown in the middle right of the non-rejection image.

can remove the sag from a set of conductors as seen in Figures 7.4 and 7.5. All of the points within the conductor search region have the estimated sag subtracted from their position when projected onto the clustering plane, See Figure 7.6.

The sag estimation model has a second use as an indicator as to whether two poles share any conductors. This is done by applying a series of rejection rules to the sag model; as was done with fitted conductor models discussed in the previous chapter. In fact, the same two rules are applied to check the sag model has a valid shape (hanging downward) and the two ends of the sag model do not extend more than 0.5 meters above their respective pole tops. In addition to these, a third rule is used to check the lowest point of the sag model does not hang closer than 2.5 meters from the ground. The ground is defined by a vector between the two pole bottoms. The threshold of 2.5 meters was chosen because under normal conditions no conductor should be closer than twice that distance from the ground. But 2.5 meters is higher than the majority of the ground clutter within close proximity of the conductors, thus making it a good test as to whether ground clutter has been incorporated into the sag model. Any two poles which are linked by a sag model which does not pass all three of these tests is rejected from the

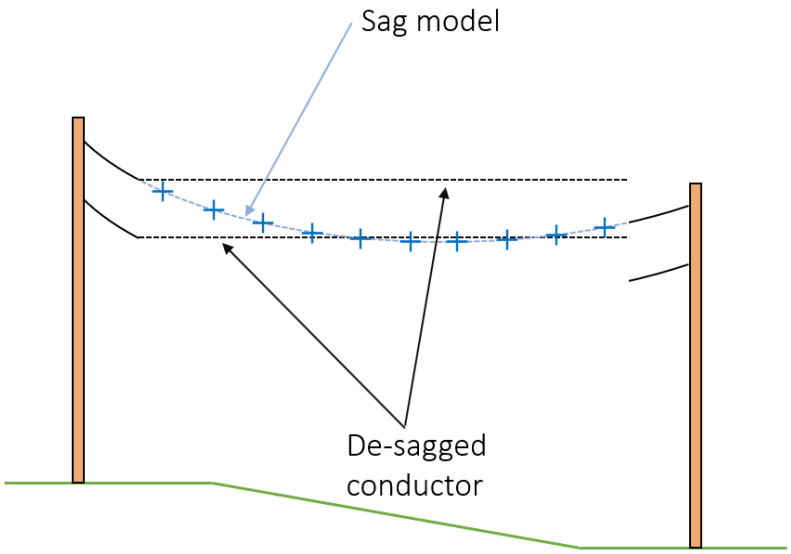


Figure 7.4: The span sag estimation model (blue dashed) can then be used to generate the de-sagged conductors (black dashed).

conductor clustering process.

7.2.2 Clustering

The CSDC method discussed previously used a 1-dimensional mean shift clustering algorithm to identify conductors on the clustering plane. With the sag compensation decreasing the vertical spread of the conductors on the clustering plane, 2-Dimensional clustering methods can now be used. While Mean Shift clustering is the preferred clustering method, K-Means and DBSCAN are also compared.

Mean Shift Clustering

Mean shift clustering essentially tries to cluster points based on an estimate of the cluster size. Through the use of a kernel function, the density of points around each point is summed, and the point is shifted based on this mean. This is repeated for all points until the system converges. If the kernel function is too small conductors will be subdivided; too large and conductors

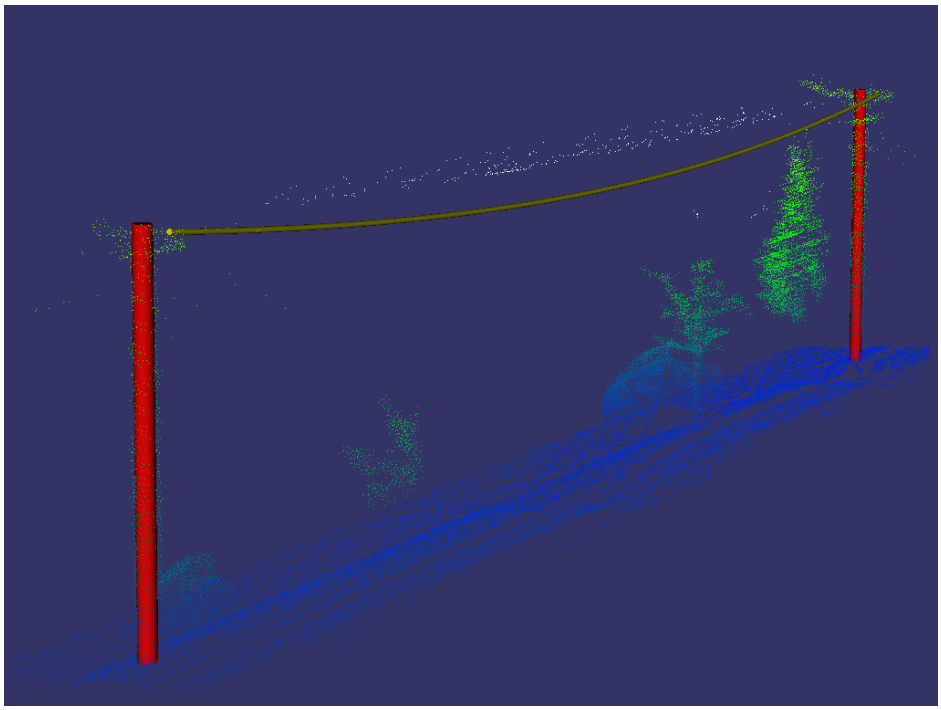


Figure 7.5: An example of a sag estimation model (yellow) and de-sagged points (white).

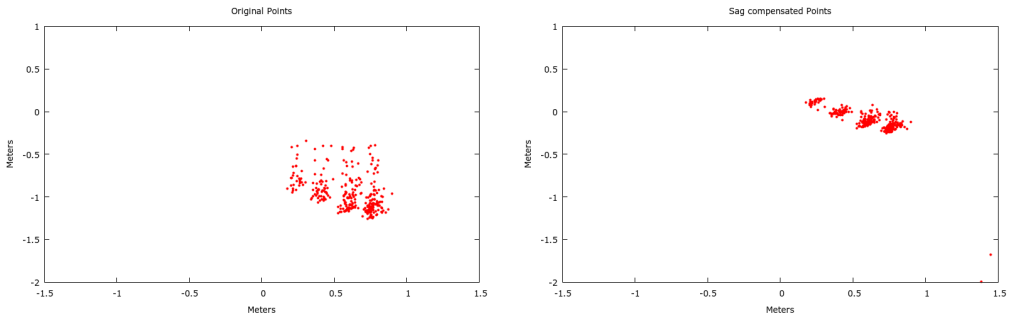


Figure 7.6: Comparison between original points projected onto the clustering plane (left) and with sag compensation (right).

will be combined. In this implementation, a Gaussian kernel function is used with a bandwidth of 4 cm; the same as was used in the CSDC algorithm.

K-Means Clustering

K-Means clustering bares some similarity to that of the mean shift clustering method. However, rather than seeding a mean with every point as with the mean shift clustering, K means is used; where K is the expected number of clusters within the dataset. Also unlike with mean shift clustering, a data point can only belong to a single mean, the closest; which has the effect of a repelling force between the means. The algorithm is considered converged when further iterations result in no movement of the means.

The K-Means algorithm is sensitive to the initialisation of the means. The means can be randomly initialised, but in this implementation, the K-Means++ algorithm is used [73].

One significant limitation of the K-Means algorithm is the requirement that the number of clusters needs to be known ahead of time. This is problematic for conductor recovery as this is not known. However, this can be overcome by running the algorithm with a range of values for K and then choosing the best result. This is done by taking the sum squared error for each run and choosing the K value using the elbow method; the point where adding more means results in diminishing returns.

DBSCAN Clustering

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is significantly different from K-Means and Mean Shift. Instead of using moving means to cluster points, DBSCAN uses a region growing method. DBSCAN clusters based on point density and uses two variables to define this behaviour; minimum number of points, and a radius. Points with more than the minimum number of neighbours within the search radius are core points, which are used to grow the region. Points on the edge of a cluster with less than the minimum number of points but reachable from a core point are considered boundary points. Other points with too few neighbours and not within the range of a core point are considered outliers. DBSCAN is the only

clustering method here to explicitly label outliers.

DBSCAN does require the minimum cluster density to be defined before starting. A trait that could be problematic because not all conductors are equally well represented in the conductor search space. Higher voltage conductors usually have a larger diameter which results in more laser returns during the scanning process.

7.2.3 Cluster Rejection

One common failure mode of the CSDC method was producing clusters of non-conductor points that appeared within the conductor search space. To reduce the presence of these false positives a post-clustering filter is applied to each of the candidate conductor clusters. This filter divides the conductor search space into three equal regions along the spanwise direction. Any clusters that do not have points within all three-thirds are rejected. The use of three regions was chosen because it is the minimum number that can be used, and thus results in the largest regions for checking occupancy. With two regions a clustered object near the shared boundary of the regions could be present in both, while also not being distributed along the span. Using more than three regions decreases the region size and thus increases the risk that a conductor will be rejected because it is not present within all regions.

7.3 Results

7.3.1 Sag Compensation

Tables 7.1 and 7.2 show the performance of the Sag Compensation. On the Hounslow dataset, all primary and secondary spans were identified and the sag correctly estimated. On the Gilberthorpes dataset 15 of 18 primary spans and 6 of 8 secondary were identified and correctly estimated. On both datasets, spans crossing the road were regularly missed. On the Hounslow dataset this was due to the short span length decreasing the number of windows available for sag estimation and thus increasing the likelihood that a sag model cannot be fitted. However, because the spans crossing the road in the Gilberthorpes dataset are longer, these spans were able to have sag

	Correct	Error
Primary	6	0
Secondary	3	0
Crossing	1	3

Table 7.1: Sag model recovery performance on the Hounslow dataset.

	Correct	Error
Primary	15	3
Secondary	6	2
Crossing	6	13

Table 7.2: Sag model recovery performance on the extended Gilberthorpes dataset.

	Correct	Error
Primary	16	2
Secondary	7	1
Crossing	11	8

Table 7.3: Sag model recovery performance with a large pole guard region of 4 meters on the extended Gilberthorpes dataset.

models fitted to them, but they were distorted due to pole mounted street lights hanging into the conductor search space; See Figure 7.7. To verify this behaviour the sag estimation was re-run with the conductor search space starting 4 meters from the pole instead of the 2.5 meters used previously. With this increased clearance the number of modelled road crossing spans increases from 6 to 11; See Table 7.3.

While the three previous tables show the performance of fitting a sag model to the top most conductors in a given span; lower conductors must also be analysed for spans with multilevel conductors. In some cases the sag model fitted to the top layer is also a good model for the lower level; See Figure 7.8. However in most cases the sag model does little to reduce the point spread due to sag, See Figure 7.9, and in the worst cases the sag model further increases the point spread, See Figure 7.10.

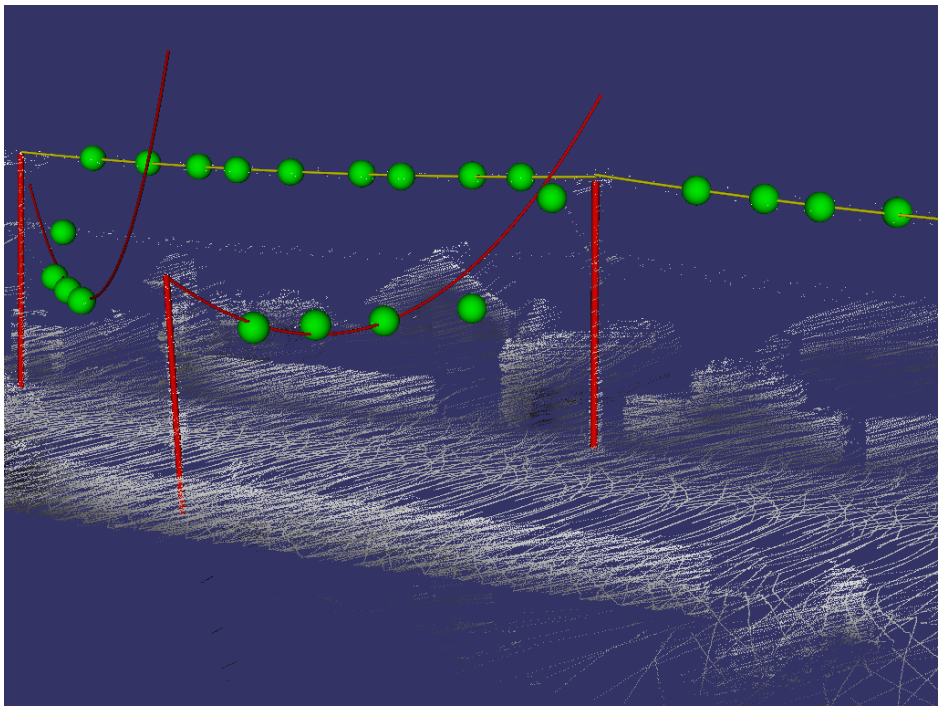


Figure 7.7: Example of sag models being distorted (red) by pole mounted street lamps hanging over the conductors.

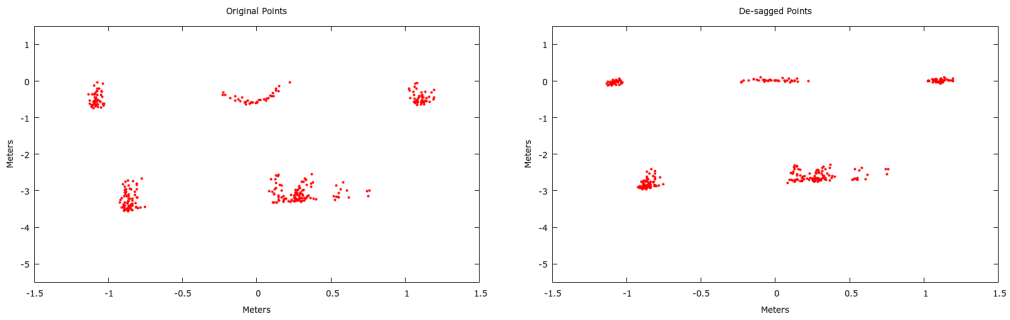


Figure 7.8: Example of span with two levels before sag compensation (left) and after (right).

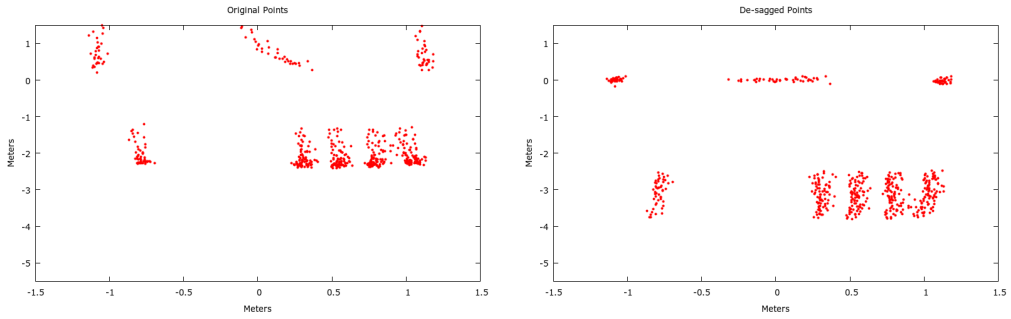


Figure 7.9: Example of sag compensation on two levels (left) only correcting the top level conductors (right).

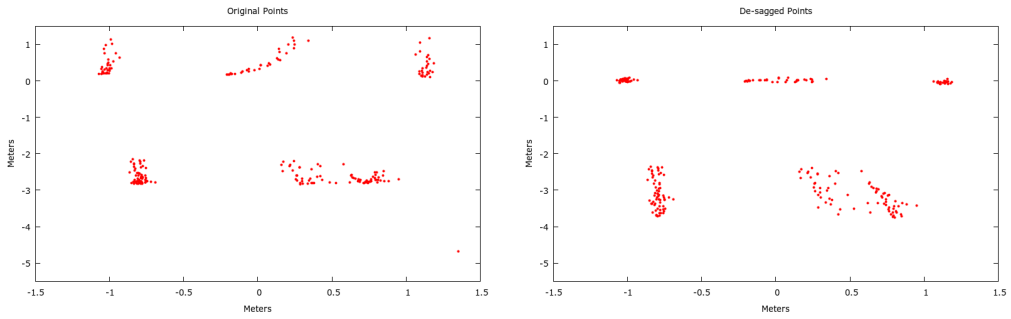


Figure 7.10: Example of sag compensation model based on top level conductors causing distortion in the lower level (right), when compared with original distribution (left).

	Accepted	Rejected
Valid Sag Model	**27**	2
Invalid Sag Model	4	**29**

Table 7.4: Sag model rejection performance on the extended Gilberthorpes dataset with a pole ignore radius of 2.5 meters

	Accepted	Rejected
Valid Sag Model	**34**	2
Invalid Sag Model	4	**13**

Table 7.5: Sag model rejection performance on the extended Gilberthorpes dataset with a pole ignore radius of 4 meters

	Accepted	Rejected
Valid Sag Model	**10**	0
Invalid Sag Model	0	**6**

Table 7.6: Sag model rejection performance on the Hounslow dataset with a pole ignore radius of 2.5 meters

7.3.2 Sag Model Rejection

The sag model rejection rules performed well on both datasets. On the Gilberthorpes dataset with both the 2.5 and 4 meter pole clearance, two sag models were rejected and four were accepted incorrectly; See Tables 7.4 and 7.5. The two that were rejected shared a common pole that was incorrectly modelled and too short. This short pole resulted in the two connected sag models terminating further above the pole than the 0.5 meter threshold. Trees distributed along the span caused the four incorrectly accepted models. On the Hounslow dataset the sag model rejection rules correctly accepted and rejected all sag models; See Table 7.6.

7.3.3 K-Means Clustering

The K-Means clustering algorithm performed the worst of the three tested clustering algorithms. Figure 7.12 shows the typical output for spans within the Gilberthorpes dataset. While the K-Means++ algorithm does attempt to spread out the initial centroids, the large distances between conductors

on the top layer and between the layers results in the centroids being unable to migrate between conductors. Conductors on the lower level tend to be closer together allowing for the movement of centroids, but due to their lower number of points compared to the upper level, K-Means++ did not tend to leave enough centroids within proximity to represent each of the conductors.

The other difficulty with K-Means is determining the number of means to use. Figure 7.11 shows the sum squared error (SSE) for a single two-levelled span with a range of values for K and both random and KMeans++ initialisation. The correct number of clusters is eight, but the SSE does not reliably fall until nine centroids are used. Even with nine centroids, K-Means can still fail to converge optimally as shown by the largest SSE values.

7.3.4 DBSCAN Clustering

The DBSCAN clustering performed significantly better than K-Means. On the Hounslow data 2/3 of the conductors were successfully clustered when an epsilon of 4cm and minimum neighbours of 10 was used; See Table 7.7. DBSCAN also performed well on the two levelled spans in the Gilberthorpes dataset; See Table 7.8. DBSCAN was able to cluster all top-level conductors and up to half on the conductors on the lower level; a better result than Mean Shift (discussed below) and despite the poor sag compensation of the lower level.

What is not immediately evident from Table 7.8, is that the relatively flat performance of the lower layer is mostly due to a trade-off when selecting parameters. With a large ε and small $minpts$, less dense clusters are more likely to be recovered, but neighbouring clusters are more likely to be merged. With the opposite parameters, smaller ε and larger $minpts$, close dense neighbours can be segmented at the cost of losing less dense and smaller clusters.

7.3.5 Mean Shift Clustering

The Mean Shift clustering algorithm was tested using a Gaussian kernel with a bandwidth ranging from 3 to 8 cm. The Mean Shift algorithm was able to match the performance of DBSCAN on the top level conductors

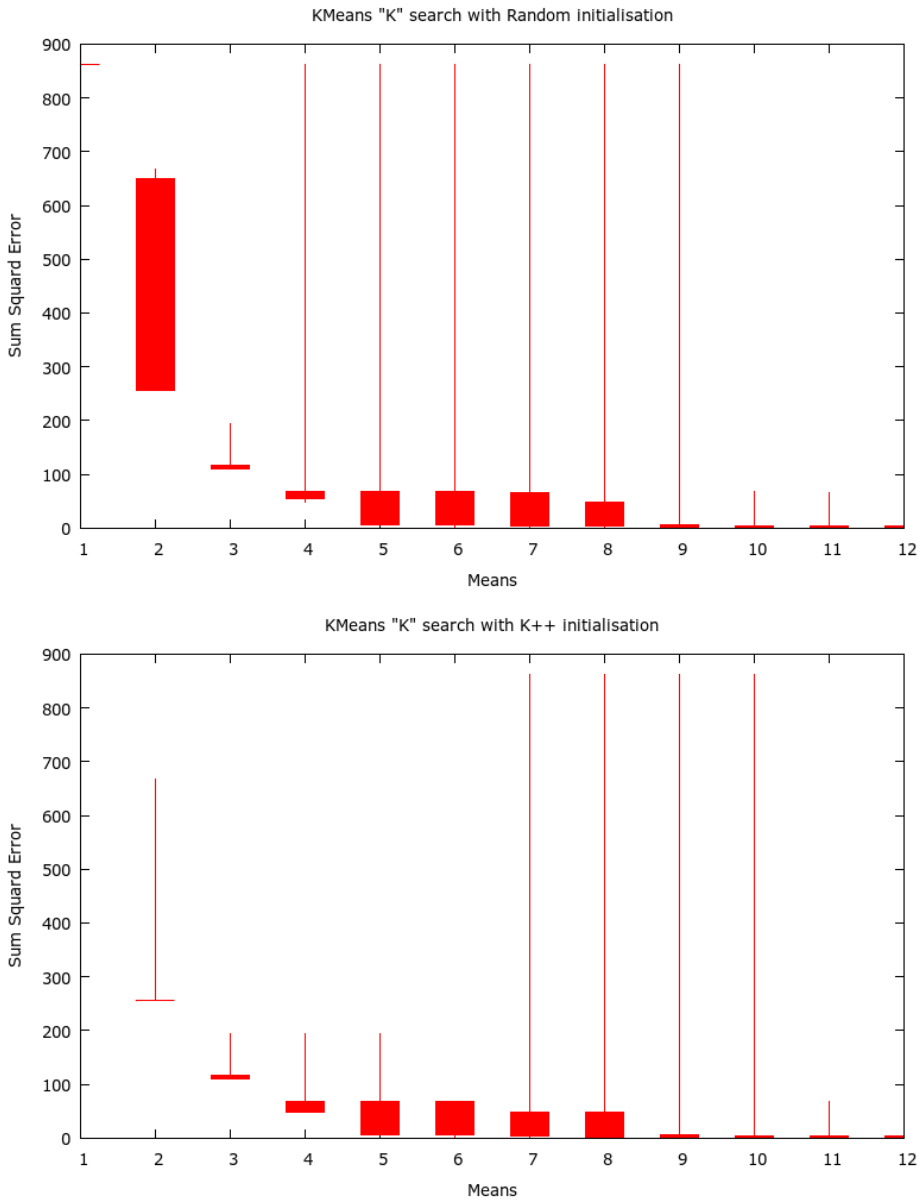


Figure 7.11: Comparison between Random and KMeans++ initialisation when searching for the optimal mean count using the Sum Squared Error metric. The data used in this example contained three conductors on the upper level and five on the lower. An example of the centroids can be seen in Figure 7.12.

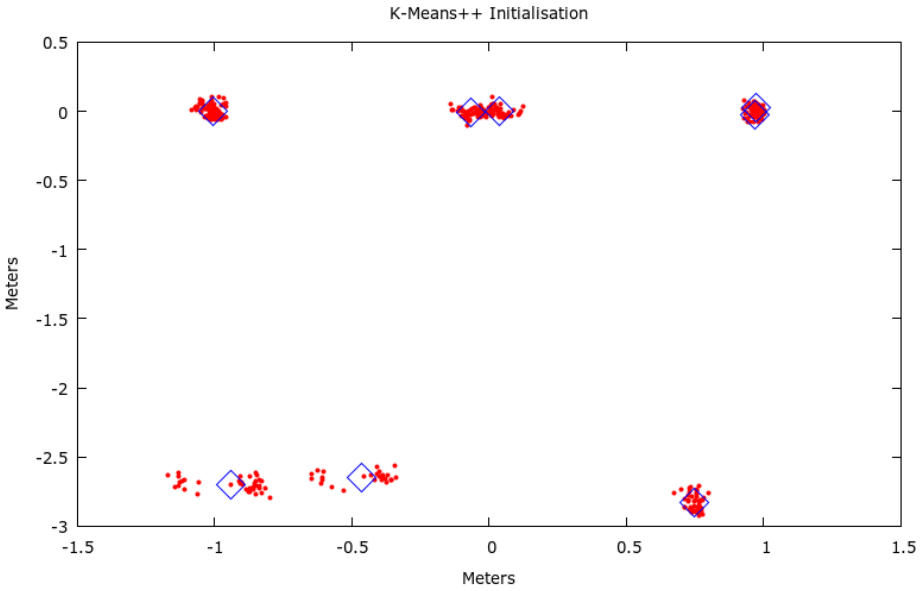


Figure 7.12: A typical example of KMeans centroids after becoming stuck at a local minimum. KMeans was explicitly given the correct number of means and was initialised using KMeans++.

	Min Points			
ε	10	20	30	40
3cm	16	13	10	10
4cm	19	14	11	10
5cm	16	12	11	6
6cm	12	13	9	8

Table 7.7: DBSCAN parameter search on the Hounslow dataset. Values are the number of conductors represented by at least one cluster which is not shared with another conductor. There are 28 possible conductors.

	Min Points			
ε	10	20	30	40
3cm	39 / 27	38 / 21	28 / 17	18 / 11
4cm	39 / 20	37 / 25	33 / 20	20 / 14
5cm	39 / 15	37 / 17	34 / 17	22 / 17
6cm	39 / 15	37 / 13	35 / 13	23 / 11

Table 7.8: DBSCAN parameter search on the Gilberthorpes dataset. Values are the number of valid clusters on upper level over the number of valid clusters on the lower level. There are 39 conductors on the upper level and 58 on the lower.

Bandwidth	Top Layer	Bottom Layer
3cm	34	13
4cm	36	17
5cm	38	21
6cm	38	21
7cm	38	21
8cm	39	15

Table 7.9: This table shows the number of successfully clustered conductors using Mean Shift Clustering with various bandwidths. Conductor spans with two layers from the extended Gilberthorpes dataset were tested.

in the Gilberthorpes dataset with kernels sizes above 4 cm. On the lower levels Means Shift clustering was consistently better than DBSCAN with kernel sizes from 5 to 7 cm; See Table 7.9. On the Hounslow dataset all but one primary conductor was clustered with kernel sizes of 4 and 5 cm. All secondary conductors except those crossing the road were clustered for kernel sizes of 5 and 6 cm. These 2-Dimensional kernels are all close in bandwidth to the 4 cm used with the 1-Dimensional kernel in the CSDC algorithm.

The Means Shift clustering algorithm was more consistent than DBSCAN at recovering the lower level conductors on the Gilberthorpes dataset. The performance of the Mean Shift clustering algorithm is highly dependant on how successful the sag compensation was. When the sag compensation performed well, the lower conductors were all correctly clustered. However, as discussed above often the lower conductors are not well sag compensated, and in these cases, the Mean Shift clustering algorithm tends to split the conductors into multiple clusters vertically; See Figure 7.13.

7.3.6 Successfully Modelled Conductors

After clustering and model fitting all conductors undergo the rejection tests. The importance of this can be seen in Figure 7.14, which shows a large number of non-conductor clusters which would otherwise result in conductors being erroneously classified.

These non-rejected conductor models are the final output from the proposed CCSC method. Table 7.11 shows the conductor recovery performance

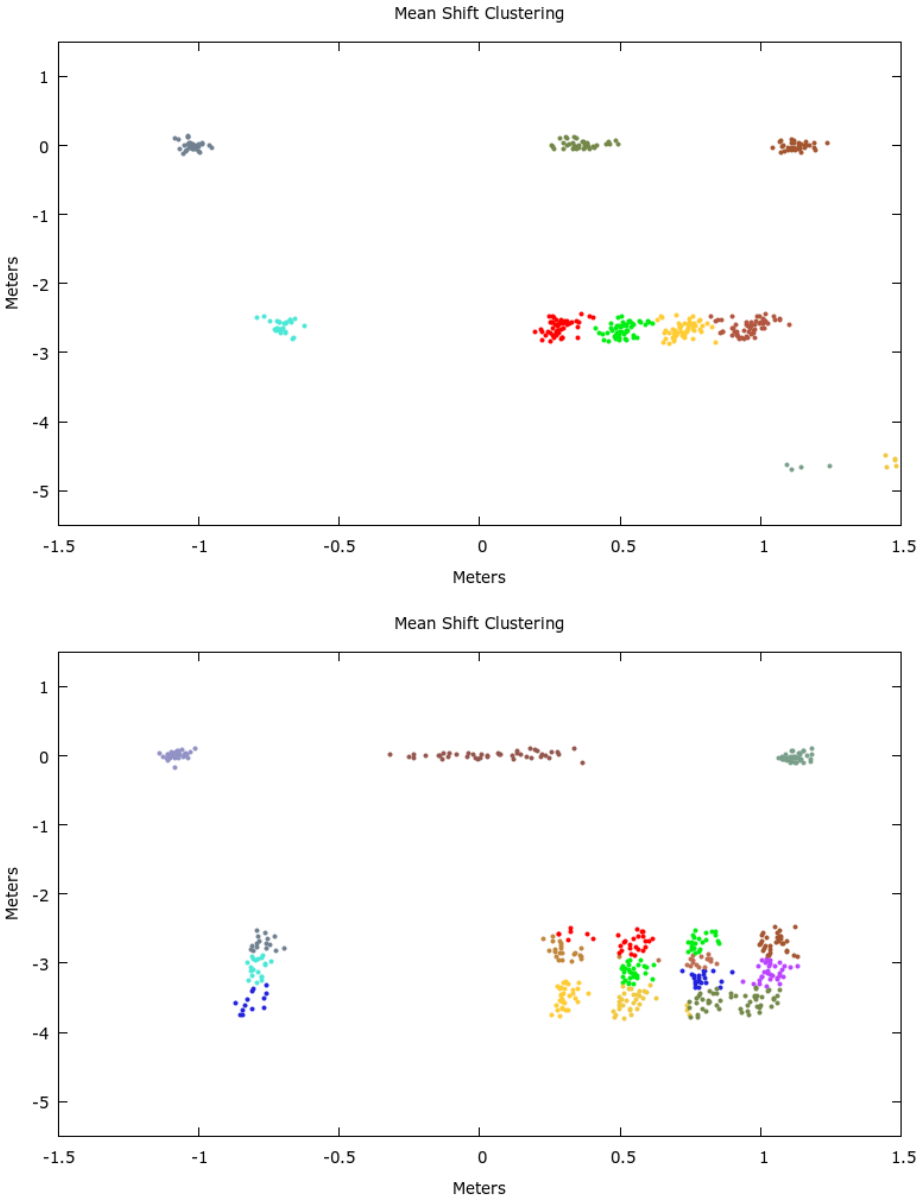


Figure 7.13: Two examples of Mean Shift Clustering with a bandwidth of 8cm. The upper figure shows the successful clustering of the lower conductor level while the lower figure shows the most common failure mode.

Bandwidth	Primary	Secondary
3cm	20	3
4cm	21	3
5cm	21	4
6cm	19	4
7cm	15	3
8cm	6	3

Table 7.10: This table shows the number of successfully clustered conductors using Mean Shift Clustering with various bandwidths. All conductors spans from the Hounslow dataset were tested.

for the Hounslow dataset consisting of only single layer spans. Of the five missed conductors only one was a conductor of primary interest and the remaining conductors were crossing the street. The primary conductor was lost during the clustering phase and the four crossing the street belonged to a span that did not contain enough occupied sag estimation windows to a have a sag model fitted.

For the Gilberthorpes dataset the conductors have been further broken down into conductor type and importance; See Table 7.12. Half of the primary (labelled top and bottom) and secondary conductors were successfully recovered. However, less than one-sixth of conductors crossing the road were recovered.

For comparison, the final set of conductors is shown in Table 7.13 using DBSCAN instead of Mean Shift Clustering. Using DBSCAN no secondary conductors or conductors crossing the road are successfully recovered. However, DBSCAN does result in 50% more lower level conductors being recovered than the Mean Shift algorithm does. The leading cause for the complete failure to recover secondary conductors and those crossing the road is their significantly reduced representation in the point cloud resulting in a much lower point density.

7.4 Discussion

Through the use of a sag estimation model, the clustering method is now able to segment conductors with no horizontal separation. Moreover, this method

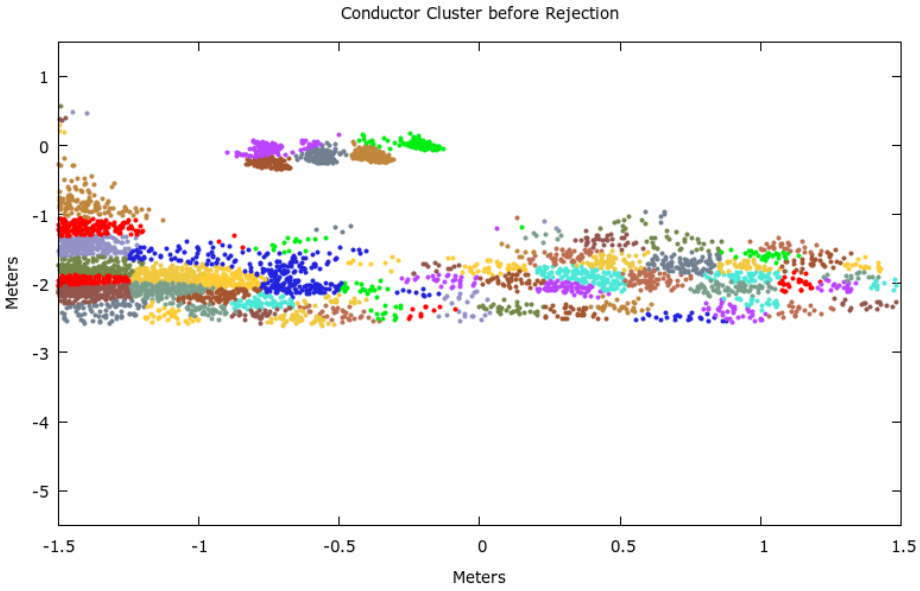


Figure 7.14: Example of many erroneous conductor clusters prior to cluster rejection. These clusters were generated with the Mean Shift algorithm from a span within the Hounslow dataset.

	Conductor	Non-Conductor
Modelled Conductors	**26**	3
Missed Conductors	5	

Table 7.11: Final output on Hounslow dataset using Mean Shift Clustering with a bandwidth of 4cm. Of the five missed, four were crossing the street, and one was in a span with three other neighbouring conductors.

	Top	Bottom	Crossing	Secondary
Modelled Conductors	**29**	**27**	**3**	**5**
Missed Conductors	25	54	16	5

Table 7.12: Final output on extended Gilberthorpes dataset using Mean Shift Clustering with a bandwidth of 4cm.

	Top	Bottom	Crossing	Secondary
Modelled Conductors	**26**	**41**	**0**	**0**
Missed Conductors	28	40	19	8

Table 7.13: Final output on extended Gilberthorpes dataset using DBSCAN with a bandwidth of 4cm.

can operate with ground clutter and trees within the conductor search space; something that was not possible with the previous CSDC method. This flexibility removes the previous requirement that the lower boundary of the conductor search space is defined before clustering; a value that needed to be changed depending on the number of conductor layers.

The sag estimation model helps reduce the spread of points due to the shape of the conductors; allowing for clustering in 2-Dimensions. However, the previous assumption that a conductor sag model based on the upper level of conductors could also be used to remove sag from lower conductors was proven false. This finding was the primary reason that conductors on the upper level were significantly more likely to be recovered than those on the lower level in the Gilberthorpes dataset.

The sag estimation model also allows for the rejection of an entire span without recovering each of the conductors and processing them individually. This rejection is done by applying similar tests to those used to check a conductor for validity. These tests both decrease the processing time by skipping additional steps and also reduce the chances of falsely fitting conductor models to spans without any conductors. These sag model rejections performed very well, and there was only one situation where a pole with an incorrectly recovered height caused both connected spans to be falsely rejected.

The three clustering algorithms K-Means, DBSCAN, and Mean Shift have been compared. Both DBSCAN and Means Shift clustering algorithms performed well while the K-Means algorithm struggled to produce decent results. The poor performance of K-Means is mostly due to the significant gaps between some of the conductors which stops the centroids from migrating between them. Even with the use of the K-Means++ initialisation algorithm the performance was not improved enough to make the algorithm a viable clustering method. Perhaps a more deterministic version of K-Means++ which removed the random element and purely selected the location with the maximum residual probability could perform better. The other challenge with K-Means is the need to determine the number of clusters in the dataset. One of the conventional methods of achieving this is the elbow method, but as can be seen in Figure 7.11, there is no reliable elbow on these datasets.

The DBSCAN clustering algorithm performed comparatively well with the Mean Shift algorithm. However, DBSCAN did fail to recover any of the secondary conductors or conductors crossing the road. Both of these sets of conductors are less represented than the primary conductors in the point cloud; either because they were further from the LiDAR in the case of the secondary, or because they were only briefly observed for the road crossing conductors. This lower representation in the point cloud reduces the density of points for those conductors. As DBSCAN operates based on point density this is problematic. To overcome this limitation, the acceptable point density could be reduced at the cost of increasing the number of false positives. It also bears keeping in mind that there will be more extreme conductor point densities than are contained within these datasets, longer spans, thicker conductors, and changing materials will all affect point density and reduces the reliability of the DBSCAN clustering algorithm.

Moving the Mean Shift clustering algorithm from 1-Dimension in the CSDC algorithm into 2-Dimensions did reduce the quality of the conductor clusters. While the top level conductors in the Gilberthorpes dataset and single level conductors in the Hounslow dataset were comparatively well recovered, sag compensated conductors on lower levels were not. This is to be expected because Mean Shift essentially tries to cluster based on the shape of a region of points. Without the correct sag compensation of the lower conductors, the shapes of these clusters are distorted. With correct sag compensation of these lower layers the Mean Shift algorithm can be expected to perform better; as indicated in Figure 7.13.

Finally, an additional conductor rejection rule was added. By checking if the points of a conductor are well distributed along the span, clusters formed by trees and other objects within the conductor search space can be rejected. This is particularly important as the lower boundary of the conductor search space is no longer manually defined and this causes a large number of non-conductor clusters to be created; See Figure 7.14.

Chapter VIII

Proposed Method for Conductor Clustering using Multiple Sag Models

8.1 Introduction

The Conductor Clustering using Multiple Sag-Models (CCMSM) method is designed to address the primary shortcoming of the previously discussed Conductor Clustering using Sag Compensation (CCSC) method; poor sag compensation of lower levels for spans with multiple levels of conductors. The CCMSM processes each conductor level individually instead of using a single sag model for all levels.

8.2 Methodology

The Conductor Clustering using Multiple Sag-Models method, like the previous CCSC and CSDC methods, operates within the conductor search space; a cuboid region bounded at opposite ends by the utility poles and 3 meters wide. However, the significant difference between the CCMSM and previous methods is the CCMSM attempts to determine the number of conductor levels before projecting points onto the clustering plane. By identifying the number of conductor levels within a span, the conductor search space can be more intelligently divided up to avoid including non-conductor points during the clustering phase, and to fit conductor sag models to individual levels.

8.2.1 Conductor levels determination

The conductor levels determination is an extension of the CCSC sag estimation step. Like the CCSC method the span is divided up along the span-wise

direction into a set of windows, and once again these windows each extend 2.5 meters along the span.

Within each window, the uppermost point is found and this defines the top of a subregion referred to as a pane. The bottom of this pane is 0.5 meters below the top. The centre of mass of the points within this pane is then found. The pane size in the CCMSM is twice that of the CCSC. The reason the pane size was increased is that even though the conductors are still not expected to fall more than 0.25 meters per window, there can be some vertical separation between conductors on the same level towards the centre of the span. It is important that all conductors on the same level are contained within the same pane; thus justifying the pane height increase.

In the previous CCSC method, it was these centre of masses that were used to fit the sag estimation model, and this was the end of the window processing. However, in the CCMSM method, we begin looking for a new pane directly below the last. This process is continued until all of the points have been included into a pane (See Figure 8.1). The centre of mass for each pane is then found; an example of this can be seen in Figure 8.2.

This process of dividing the window into panes and finding the individual centre of masses solves two significant challenges the CCSC method had. With the CCSC it was shown that increasing the pane size to include multiple conductor layers resulted in unstable centre of masses as they were pulled towards the conductor with the most points within a given window. The other reason the pane could not be extended to include more than one level of conductors is the number of conductor levels was not known. With this process of dividing the window into multiple panes, both of these challenges are overcome. Each conductor level is represented within each window with its own centre of mass, and the height of each pane is small enough not to include the points of any objects below.

The centre of masses within each window are then filtered (See Figure 8.4). The first filter is a proximity-based filter. Any centre of masses within a window which have less than 0.60 meters vertical separation are deleted. The threshold of 0.60 meters was chosen because it is 0.1 meters more than the vertical size of a pane. If two panes are stacked closely and their centre of masses fall on the same object then the average vertical separation should

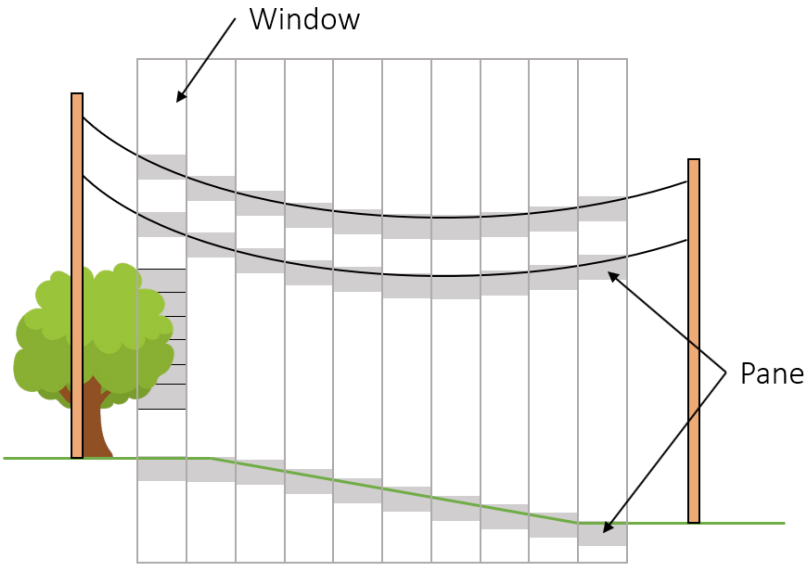


Figure 8.1: Each window is traversed downwards and new panes are fitted until all points within the window are also included within a pane.

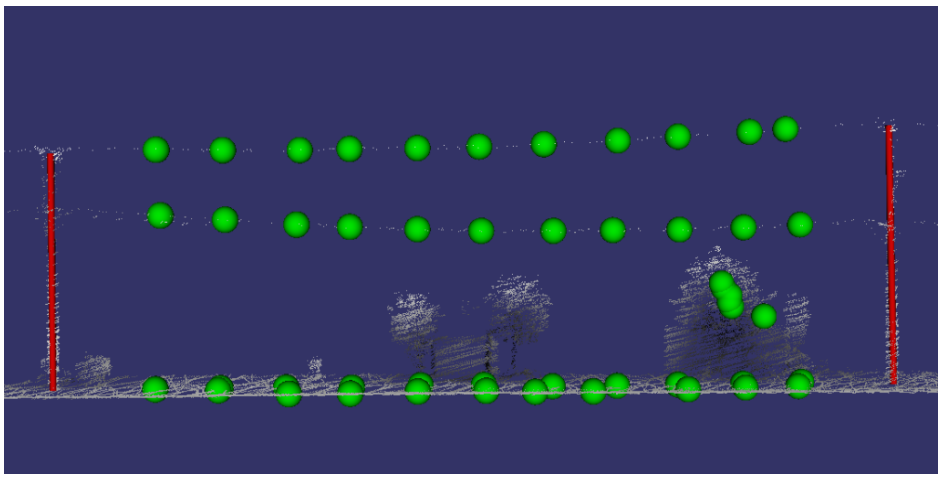


Figure 8.2: An example of the multiple centre of masses found for each window. Note the bush in the lower right is partially within the conductor search region and thus has centre of masses associated with it.

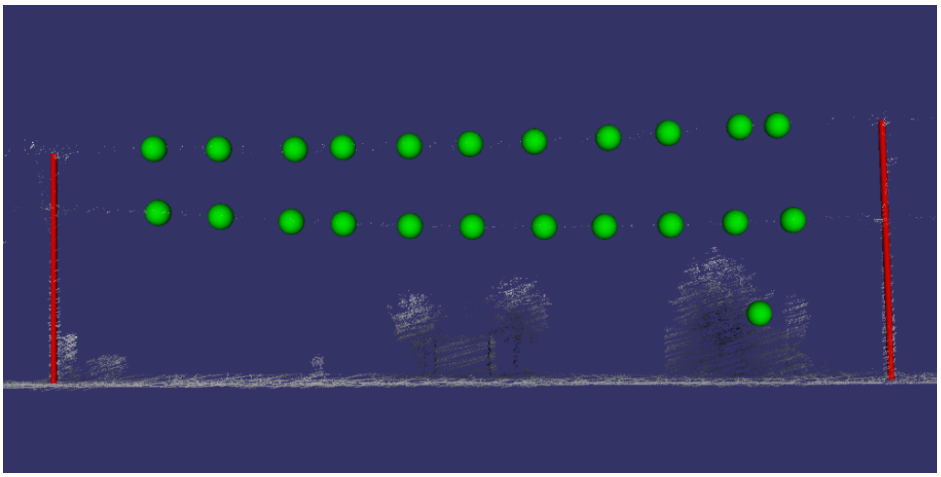


Figure 8.3: The remaining centre of masses after filtering. Note all the centre of masses belonging to the ground and the majority of the bush have been removed. The single bush centre of mass remaining fell in a separate window to the rest of the object.

be 0.5 meters.

This filter is designed to exploit the geometry of the conductors. Because the conductor levels are thin, mostly horizontal structures and are contained within a single pane, they will be represented by one centre of mass. Non-conductor objects within the conductor search space will likely have more than 0.5 meters in height and thus will span multiple panes, and have multiple close centre of masses. It is these close centre of masses this filter removes.

The second filter removes any centre of masses less than 1 meter above the vector linking the two pole bottoms. This filter is designed to remove points along the ground. This second filter is required because often the ground will have less than 0.5 meters in height difference within a single window. With such a small variance in height, the ground will occupy a single pane and result in only one centre of mass. The first proximity based filter will not filter this single centre of mass along the ground and thus this second filter is required.

It is important the filters be applied in the described order. By applying the ground removal filter first, short objects may have most of their centre of masses removed. It is conceivable that centre of masses towards the top of their structure may have their neighbours culled and thus pass the proximity

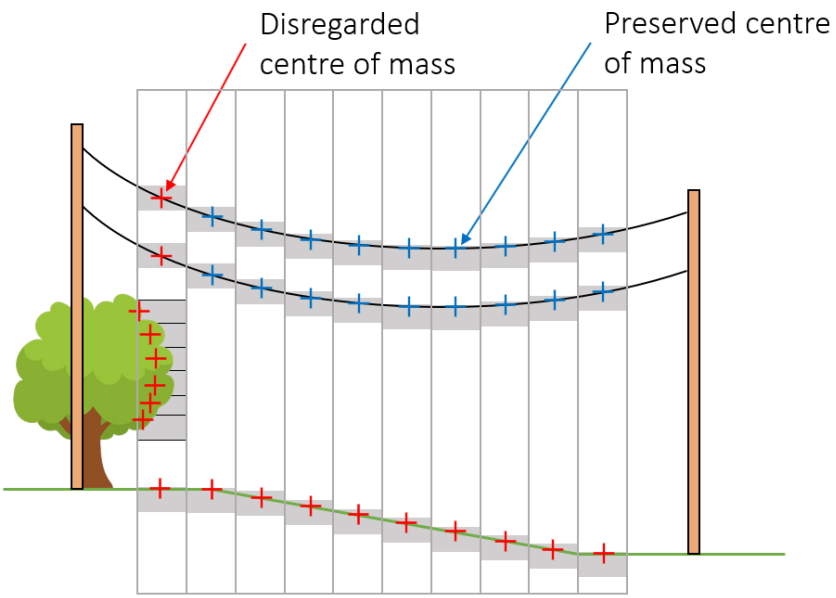


Figure 8.4: Example showing filtered centre of masses. COMs along the ground are filtered due to their low height, those belonging to the tree are filtered due to their close proximity to each other. The left most COMs on the conductors are filtered because they belong to a window with the incorrect number of COMs.

check. Figure 8.3 shows the filtered centre of masses from the two-layered span originally shown in Figure 8.2.

Finally, the number of conductor levels can be found by taking the mode of the number of centre of masses within each window. In the case presented in figure 8.2 there are 10 windows with 2 centre of masses and 1 window with 3. So the number of conductor levels is correctly found to be 2.

8.2.2 Sag Compensation

Now with the filtered set of centre of masses and the number of conductor levels known, it is now possible to fit the conductor sag models. The first step is to group the centre of masses into collections based on their vertical order within each window, i.e. all of the topmost centre of masses are grouped together. The number of collections is set to the number of expected levels. Any centre of masses which belong to a window with an incorrect number of

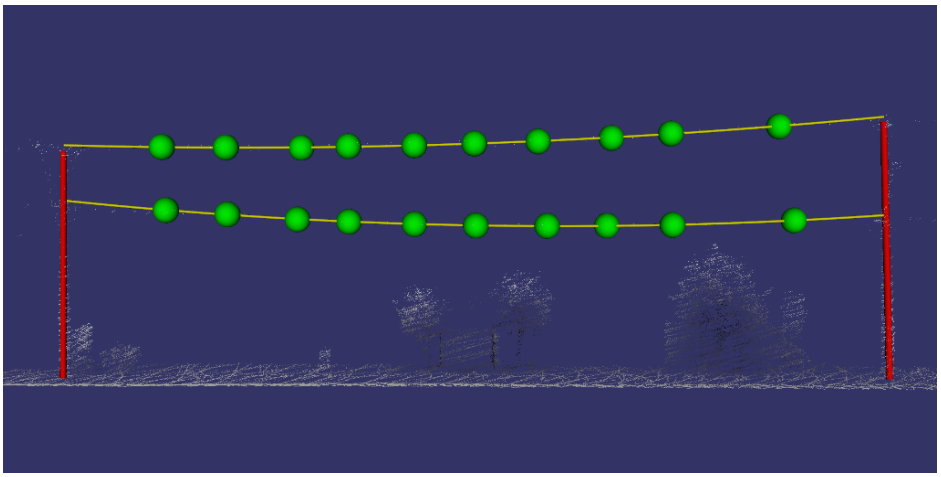


Figure 8.5: Example of two fitted sag models and the centre of masses used for fitting. Note the centre of mass that was part of a bush in the lower right along with the two directly above in the same window are no longer present when compared with figure 8.3

centre of masses are not included within the collections.

The decision to disregard all centre masses within a window with an incorrect number of centre of masses was chosen because it is difficult to identify those of which are invalid. If there are too few centre of masses the conductor level missing is not known. Likewise, with too many centre of masses, the identity of the extra centre of mass is also not known. It is feasible that the validity of a centre of masses could be inferred by looking at an adjacent window. But this requires that we make the assumption that any anomaly in one window is not present in a neighbouring one; likely to be an unsafe assumption.

Using these collections of centre of masses a conductor sag model can be fitted for each level. This fitting process is similar to sag model fitting in the CCSC method with the exception that the centre of masses have been filtered and there is a high certainty that all of the centre of masses within a collection belong to the same conductor. In figure 8.5 an example of two fitted sag models is shown. Note there are two distinctly different curves for the two levels; something the CCSC method was unable to describe.

The fitted sag models have the same validity tests applied as in the CCSC method. The sag models are checked to be hanging under gravity and that

they don't hang closer than 2.5 meters from the ground. The ends of the sag model are tested to ensure they don't terminate more than 1 meter above the pole tops. In the CCSC method this threshold was set to 0.5 meters and resulted in incorrect rejections; for this reason, it has been extended to 1 meter.

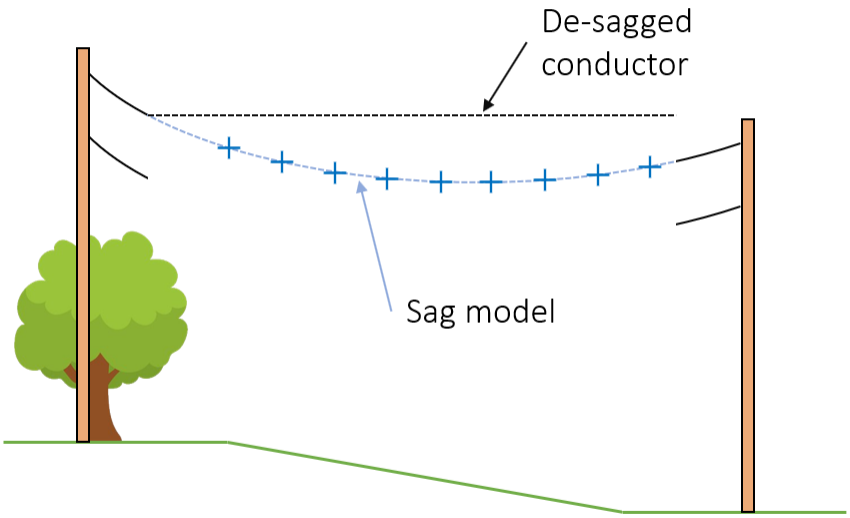
The sag models can now be used to remove the estimated sag from the conductors as the points are projected onto the clustering plane. In the two previous methods, CSDC and CCSC, all points above a given height were projected onto the clustering plane. With CCMSM the projection of points onto the clustering plane can be more intelligently executed. Instead of using a single clustering plane, a clustering plane is created for each sag model. For each clustering plane, only the points with less than 0.25 meters vertical distance from the associated sag model are projected onto the plane (See Figure 8.6). With this approach, only points from a single conductor level will be found on any given clustering plane. It is also unlikely that non-conductor points will be projected onto the clustering plane because the conductors should always have more than 0.25 meters of vertical clearance. Figure 8.7 shows an example of the two clustering planes created from the two levelled span referenced throughout this section.

8.2.3 Clustering

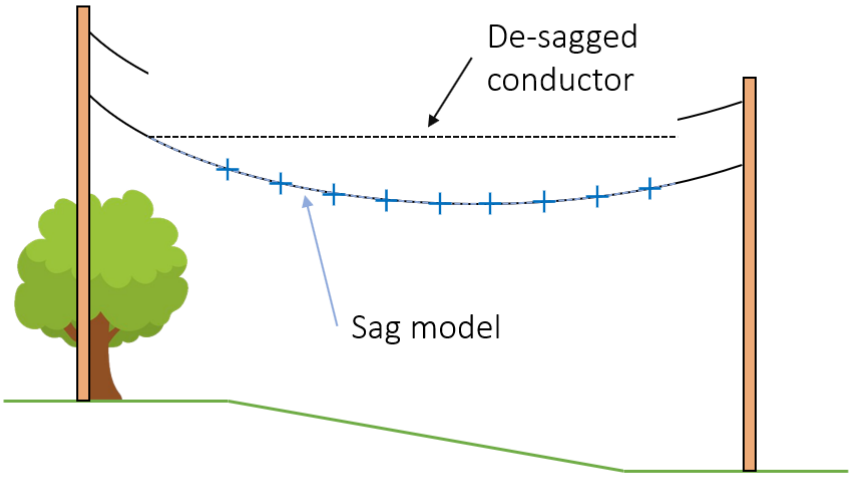
Clustering within the CCMSM method is significantly simpler than in the two previous methods. This is because the conductors are better sag corrected and there is generally no non-conductor points on the clustering plane. Either DBSCAN or Mean Shift could be used to perform 2-Dimensional clustering on the clustering plane. Interestingly, the 1-Dimensional Means Shift clustering algorithm used in the CSDC method can also be used because only a single layer of conductors will be found on each clustering plane.

8.2.4 Cluster Rejection

The need for cluster rejection is reduced because the CCMSM method attempts to avoid projecting non-conductor points onto the clustering plane. However, it can be re-purposed to reject smaller satellite clusters around

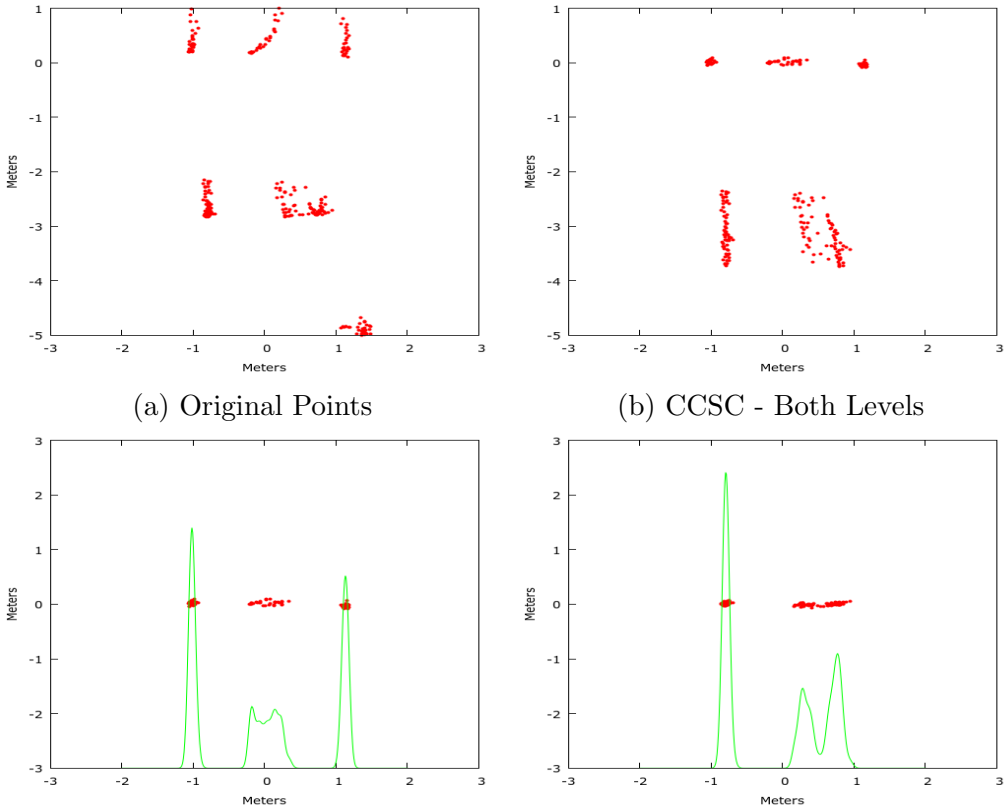


(a) De-sagged upper conductor



(b) De-sagged lower conductor

Figure 8.6: Using the individual sag models, each level can be individually de-sagged and projected onto the clustering plane.



(a) Original Points

(b) CCSC - Both Levels

(c) CCMSM - Upper Level

(d) CCMSM - Lower Level

Figure 8.7: These figures show the projection of points onto the clustering plane for the span featured in Figures 8.3 and 8.5. Figure (8.7a) shows the points projected without sag removal, the points in the lower right belong to a nearby tree. Figure (8.7b) shows a the projected points using a single sag model as implemented by the CCSC method. Note the spread of the points on the lower level has been increased by this model. Figures (8.7c) and (8.7d) show the upper and lower levels after sag removal using separate sag estimation models as implemented by the CCMSM method. The probability density function is plotted to help show the point density.

	Correct	Error
Primary	6	0
Secondary	3	0
Crossing	1	3

Table 8.1: Sag model recovery performance on the Hounslow dataset.

conductors which can be present if the Mean Shift clustering bandwidth is reduced. By rejecting these clusters and reducing clustering bandwidth, more precise segmentation between close clusters can be achieved. The cluster rejection used by the CCMSM method is the same as used by the CCSC method. Any clusters not present in all three thirds of the span are rejected.

8.3 Results

8.3.1 Multi-level Sag Estimation

The Multi-level sag estimation performed well on the Hounslow dataset, fitting valid sag models to all primary and secondary conductors; see Table 8.1. It performed so well that it fitted a sag model to a previously unnoticed second conductor level on a primary span at the peripheral of the scan. Only one of the four spans crossing the road was recovered. The recovered span took a longer diagonal path across the road, while the other three spans were shorter and did not occupy enough windows along the span to fit a sag model.

The Multi-level sag estimation also performed well on the extended Gilberthorpes dataset, fitting sag models to all 18 primary spans and 7 of 8 secondary spans; See Table8.2. Over half of the spans crossing the road were also fitted, primarily due to the spans being longer than those in the Hounslow dataset. In one case where two conductors were crossing the road stacked vertically, both were assigned a sag model.

8.3.2 Sag Model Rejection

The sag model rejection rules produced less false rejections than observed in the CCSC method. This was primarily due to the increased height a conduc-

	Correct	Error
Primary	18	0
Secondary	7	1
Crossing	10	9

Table 8.2: Sag model recovery performance on the extended Gilberthorpes dataset.

	Accepted	Rejected
Valid Sag Model	**11**	0
Invalid Sag Model	0	**5**

Table 8.3: Sag model rejection performance on the Hounslow dataset

	Accepted	Rejected
Valid Sag Model	**54**	0
Invalid Sag Model	10	**8**

Table 8.4: Sag model rejection performance on the extended Gilberthorpes dataset

tor is allowed to terminate above the pole top. On both the Hounslow (See Table 8.3) and Gilberthorps (See Table 8.4) datasets there were no falsely rejected sag models. On the Hounslow dataset, all incorrect sag models were rejected and 45% were rejected on the Gilberthorpes dataset.

8.3.3 1-Dimensional Mean Shift Clustering

The 1-Dimensional Means Shift clustering was able to cluster all primary conductors with a kernel size of 3 and 4cm on the Hounslow dataset; see Table 8.5. With larger kernels, the primary conductors began being clustered together due to their proximity. All secondary conductors were correctly clustered regardless of the kernel size because they all had larger clearances.

For the primary spans in the extended Gilberthorpes dataset, all top-level conductors were clustered except when using a 3cm kernel. With the smaller kernel one conductor was split into too many clusters to continue processing; See Table 8.6.

Bandwidth	Primary	Secondary
3cm	24	7
4cm	24	7
5cm	23	7
6cm	14	7
7cm	6	7
8cm	0	7

Table 8.5: This table shows the number of successfully clustered conductors using 1D Mean Shift Clustering with various bandwidths. All conductor spans from the Hounslow dataset were tested.

Bandwidth	Top Layer	Bottom Layer
3cm	53	84
4cm	54	78
5cm	54	71
6cm	54	58
7cm	54	46
8cm	54	42

Table 8.6: This table shows the number of successfully clustered conductors using 1D Mean Shift Clustering with various bandwidths. Conductor spans with two layers from the extended Gilberthorpes dataset were tested.

Bandwidth	Primary	Secondary
3cm	24	4
4cm	24	4
5cm	24	4
6cm	24	4
7cm	16	6
8cm	6	7

Table 8.7: This table shows the number of successfully clustered conductors using 2-Dimensional Mean Shift Clustering with various bandwidths. All conductors spans from the Hounslow dataset were tested.

8.3.4 2-Dimensional Mean Shift Clustering

The 2-Dimensional Mean Shift clustering performed better over a broader range of kernel sizes than its 1-Dimensional counterpart at extracting the primary conductors on the Hounslow dataset; See Table 8.7. Upon closer inspection it can be seen that while only a single level of conductors is being clustered, there is still some cross span-wise overlap between the conductors which will result in less defined boundaries between clusters when only 1-Dimension is considered; See Figure 8.8. For smaller kernel sizes, not all of the secondary conductors could be clustered.

The 2-Dimensional Mean Shift clustering was able to recover all top layer conductors for kernel sizes greater than 4cm. With a kernel size of 4 cm 82 of the 90 bottom layer conductors were successfully clustered. A steady drop off in performance in bottom layer clustering was observed for larger kernel sizes due to the merging of closer conductors; See Table 8.8.

8.3.5 Conductor Recovery Performance

As with the previous conductor recovery methods, the performance of the CCMSM was analysed by manual inspection. On the Hounslow dataset 23 of the 24 primary conductors, 2 of 3 secondary, and 1 of 4 crossing conductors were recovered; See Table 8.9. There were no false positive conductors recovered. The one primary conductor that was missed belonged to a span that was not well observed at the end of the scan; See Figure 8.9. The conductor was clustered but then rejected by the rule requiring that conductors

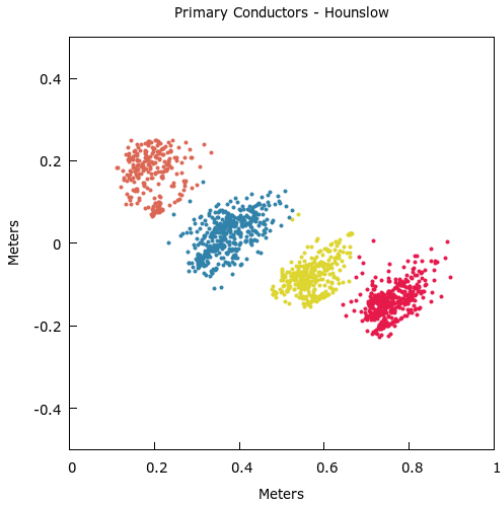


Figure 8.8: Example of 2-Dimensional Mean Shift clustering of primary conductors from the Hounslow dataset. Note there is some cross span-wise overlap which results in less defined boundaries if only the 1-Dimension is used.

Bandwidth	Top Layer	Bottom Layer
3cm	50	78
4cm	52	82
5cm	54	74
6cm	54	63
7cm	54	50
8cm	54	43

Table 8.8: This table shows the number of successfully clustered conductors from the extended Gilberthorpes dataset using 2-Dimensional Mean Shift Clustering with various bandwidths.

	Conductor	Non-Conductor
Modelled Conductors	**26**	0
Missed Conductors	5	

Table 8.9: Final output on Hounslow dataset using 2-Dimensional Mean Shift Clustering with a bandwidth of 4cm.

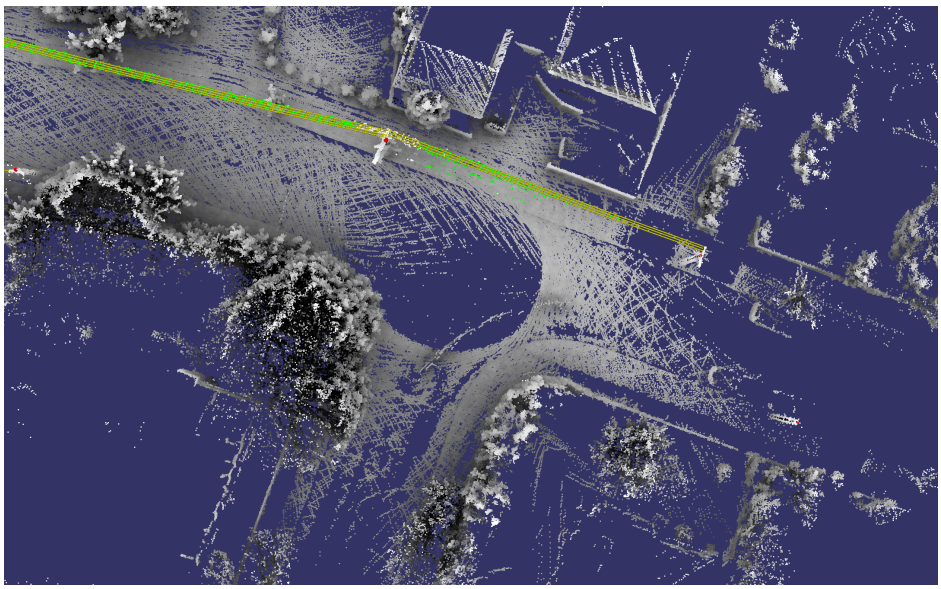


Figure 8.9: Example of a poorly observed span which caused a primary conductor to be rejected because it did not contain points along the entire span. Clustered conductor points are rendered green and fitted models are yellow.

have points distributed along the entire span. If this rule is removed the conductor can be recovered (See Figure 8.10) at the cost of losing robustness to non-conductor objects on the clustering plane.

On the extended Gilberthorpes dataset 90% of the top level conductors and 91% of the bottom level conductors were recovered from the primary spans; See Table 8.10. 25% of the conductors crossing the road and 75% of the secondary conductors were recovered. All of the missed top level conductors were the centre conductor. This centre conductor weaved from the left to the right of the poles which caused the projected points to be spread horizontally on the clustering plane; See Figure 8.11. This horizontal spread results in the conductor being broken into multiple clusters. Furthermore, the left most

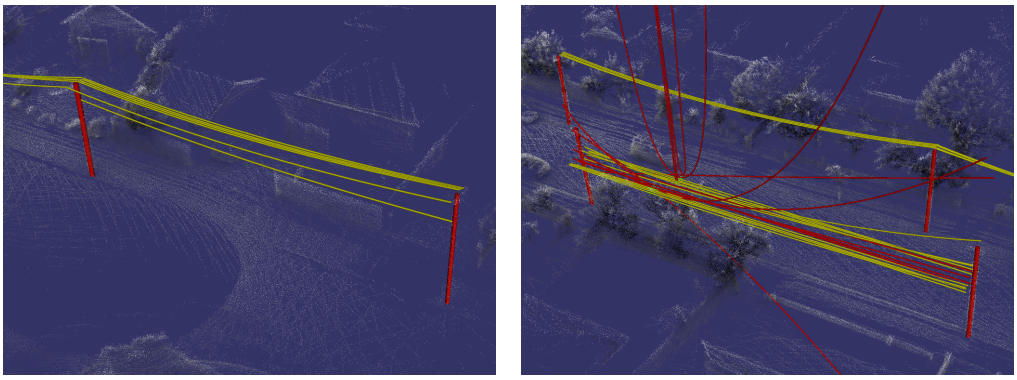


Figure 8.10: Removing the requirement that a conductor cluster must have points distributed along the entire span can increase conductor recovery in less well-observed regions (left) at the cost of less robust rejection of non-conductor objects on the clustering plane (right).

	Top	Bottom	Crossing	Secondary
Modelled Conductors	**49**	**74**	**5**	**6**
Missed Conductors	5	7	14	2

Table 8.10: Final output on extended Gilberthorpes dataset using 2-Dimensional Mean Shift Clustering with a bandwidth of 4cm.

points of these clusters will be close to one pole while the right are closer to the other. Because these clusters are more distributed towards a particular end of the span, they tend to be rejected by the rule stating clusters must be found along the entire length. It is for this reason that these top-level conductors were missed.

On the Gilberthorpes dataset, there were also three falsely recovered conductors. Two of these falsely recovered conductors were the result of two shorter conductor spans being co-linear. While both of the short spans were successfully recovered, they also produced two false positives when treated as a single larger span.

Correctness and Completeness

For comparison with Cheng et al's work on Extraction of power lines in urban environments, we have adopted their two primary metrics (8.1,8.2).

Figure 8.11: Example of the weaving centre conductor being split into two clusters due to it not being parallel to the span.

$$Completeness = \frac{TP}{TP + FN} \tag{8.1}$$

$$Correctness = \frac{TP}{TP + FP} \tag{8.2}$$

An important note of difference is Cheng used the length of the conductors while here the absolute count of conductors has been used.

	Completeness	Correctness
Hounslow - Primary	95.6 %	100%
Gilberthorpes - Top	90.7%	96.1%
Gilberthorpes - Bottom	91.4%	100%
Combined	91.8%	98.6%

Table 8.11: Completeness and Correctness for primary conductors in Hounslow and Gilberthorpes datasets.

Table 8.11 shows a break down of the completeness and correctness for the CCMSM method. These metrics where generated based on the conductor recovery rates rather than the estimated conductor length. This is because the already recovered poles constrain conductor length in the CCMSM. The

conductor count is a more reliable metric for our dataset as there is no second measurement source to provide a conductor length ground truth.

Linearity Filter Approach

The voxel-based linearity filter used in Cheng's method was implemented and tested in the Hounslow dataset, allowing for a performance comparison beyond the Completeness and Correctness values presented. The voxel-based linearity filter removes non-conductor points by dividing the point cloud into voxels and performing a linearity test on the structures within them. The linearity test first computes the three eigenvalues ($\lambda_1 > \lambda_2 > \lambda_3$) describing the structure of the points within the voxel. The linearity of the voxel is then calculated by comparing the two largest eigenvalues, as shown in Equation 8.3. The points within a voxel are disregarded if the linearity is below a set threshold; 0.3 in Cheng's work. The results of this experiment can be seen in Figure 8.12. When applying the voxel-based linearity filter, the voxel size needs to be less than the distance between conductors so that at most a single conductor is contained within a voxel; for the Hounslow dataset the voxel size should be less than 0.2 meters.

$$Linearity = (\lambda_1 - \lambda_2)/\lambda_1 \tag{8.3}$$

8.4 Discussion

In this chapter, the third iteration of our method of conductor recovery was presented. The CCMSM overcomes many of the challenges the previously presented methods struggled with, for instance, the recovery of multi-levelled conductors, the rejection of foliage within the conductor search space, and robust clustering of conductors with significantly different point densities.

When compared against state of the art in urban power line surveying using vehicle-mounted LiDAR we see a similar level of performance. In Cheng's work, a correctness of 99.1% and completeness of 93.9% was achieved. While these numbers are slightly greater than those achieved by CCMSM; 98.6% and 91.8% respectively; they fail to account for other differences between the

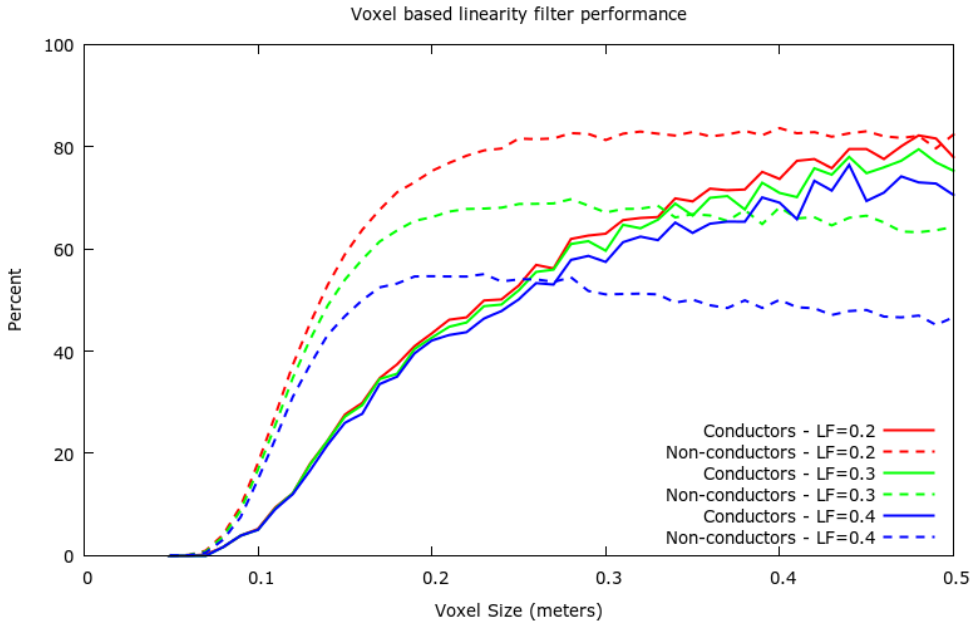


Figure 8.12: Percent of conductor and non-conductor points removed by the voxel based linearity filter performance on the Hounslow dataset as implemented by Cheng et. al.

systems. While the distances between conductors is not given in Cheng's works, imagery from the surveyed street indicates that the conductors had significantly more clearance between them than those processed by CCMSM; See Figure 8.13. Many previous methods of conductor recovery, including that presented by Cheng, rely on finding the eigenvectors to traverse along the spans. As previously stated, using eigenvectors is problematic when the conductors are close together. This is because it is difficult to include enough points to generate stable eigenvectors, while not increasing the size of the search space to where it is polluted with points from adjacent conductors. By using the concept of a clustering plane instead of conductor traversal, conductors far closer together can be recovered with comparable levels of performance.

By increasing the allowable height a sag model can terminate above a pole from 0.5 meters to 1 meter, the false rejection of sag models observed in the CCSC method is mitigated. This is important, because like many of the stages in the presented methods, once rejection has occurred, there is no

means for recovery. It is for this reason that any stage where some object is going to be rejected every effort is made to ensure a false rejection is unlikely, generally at the cost of precision. Of course, this trade-off results in more false positives being passed to the next stage. But subsequent stages can continue to apply additional filters to reduce the false positive rate further.

One early assumption that was present in all three of the conductor recover methods was that conductors run parallel to the span direction. While this mostly held true, the centre conductor on the top level of the Gilberthorpes dataset attached to the cross arm on alternating sides of the pole. This zig-zag path resulted in a distorted projection of the conductors' points onto the clustering plane. With larger clustering kernels these distorted clusters could still be recovered but at the cost of merging the lower level conductors that were closer together. Of course, different kernel sizes could be used for different conductor levels, but this would require prior knowledge of the conductor configurations. Instead, a method similar to sag compensation could be applied to each level of conductors in an attempt to measure how far off parallel the conductors are, and then subtract this difference as is done with sag.

On the Hounslow dataset one primary conductor was missed at the end of the scan where the point-cloud was less uniform and dense. The span where the conductor was missed was only well represented at one end. While the conductor was clustered, it was rejected because it didn't have points along the entire span. By disabling this rejection rule the missing primary conductor and the secondary telecommunication cable below were recovered. However, by removing this rule, other areas of the scan where trees were inside the conductor search space resulted in erroneous conductors being recovered; See Figure 8.10.

Two aligned short spans caused an unexpected failure. While the conductors belonging to the two spans were correctly recovered, there were two additional falsely recovered conductors produced by combining the two spans into one. The CCMSM method performs no checks to ensure that the geometry of the poles is valid other than measuring their distance. A possible extension would be to check a candidate span against those around it for conflicting infrastructure.

Figure 8.13: Comparison between Cheng's Nanjing dataset (left), Gilberthorpes (centre), and Hounslow (right). Note the Nanjing image is from 2014 while the LiDAR survey was conducted in 2011.

Chapter IX

Conclusion

The availability of a low-cost method to survey overhead electrical infrastructure has huge value to owners and operators of transmission networks. This thesis has identified that there is no middle ground between the high-cost airborne surveys and manually taking measurements directly. By combining an affordable vehicle-mounted LiDAR with a set of algorithms designed around the particular quirks of such sensors, a method of rapidly surveying overhead conductors at a low cost is now possible. With this technology facilitating shorter time periods between the resurveying of circuits, clearance tolerances can be more precisely defined allowing for higher utilisation of existing infrastructure through the use of Dynamic Line Rating.

Highlighted below are four distinct contributions presented within this thesis to the management of electrical networks and the field of remote sensing.

1. A method of recovering pole location and orientation with a focus on accurate pole top localisation has been presented. This method is robust to portions of the pole being unobserved or the pole shape being distorted by equipment attached to it or within close proximity. It is also resistant to falsely reconstructing poles from trees and other tall non-pole objects; a task which has proven difficult for previous methods to achieve.

2. A novel approach to power line reconstruction using a contextual understanding of the environment instead of the more traditional point-based classification. With the development of the clustering plane, conductors can be reconstructed in configurations too tightly packed for point-based features to operate effectively.

3. Through the use of sag-compensation and conductor-level segmentation, conductors can be robustly clustered environments which previously proved problematic. This is a novel method which allows the conductors first to be separated from their environment before the recovery of the individual conductors is executed.

4. Demonstrated that existing conductor recovery performance can be achieved without the use of high-cost, survey grade LiDAR equipment. There is still work to do to deploy this research, but given the results demonstrated, the resources can be procured to complete this integration.

9.1 Future Work

One of the core principles used when developing the methods described in this thesis was always to prefer false positives over false negatives. The reasoning behind this mentality was false positives can still be rejected later, while generally speaking false negatives are lost. This reasoning, however, cannot be employed in the final stages of classification and every effort must be made to label the data correctly. Further research focused on improving the precision of which reconstructed conductors are rejected could yield improvement to the system performance. Currently, conductors are rejected based on their shape and position relative to the poles and ground, however, testing the fit of the model to the underlying conductor points could prove to be a powerful feature when detecting erroneous models.

The research focus of this work was on developing methods to reconstruct electrical infrastructure from the collected point clouds. Fewer resources were invested in the development of the Data Acquisition System; consequently, there remain some areas for improvement. The integration of GPS and IMU sensors have already been discussed; but with the rapid developments in the field of autonomous vehicles, there are many newer approaches to be explored, such as the integration of monocular cameras for visual odometry and object segmentation with modern machine learning techniques. In addition to improvement of the sensors comprising the DAS, there is also the

possibility of integrating the DAS into other platforms such as unmanned aerial vehicles or a portable configuration for handheld operation.

Going forward the focus will be on integrating the methods developed in this thesis into the existing systems used for managing assets for power distribution. Throughout this research, no prior information about the location or existence of assets was provided. However, it can be expected that when augmented with an existing knowledge base containing the number, configuration, and rough location of assets, the system could run almost error-free without any human interaction.

Appendix A

LiDAR Intersection Table

Table A.1: Laser / Conductor intersection for various LIDAR inclination

		0 Degress			**5 Degrees**			**10 Degrees**			**15 Degrees**		
Beam	**Angle**	**Int**	**Bwidth**	**B/W**	**Int**	**Bwidth**	**B/W**	**Int**	**Bwidth**	**B/W**	**Int**	**Bwidth**	**B/W**
0	-15	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1	-13	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	114.67	0.37	37.25
2	-11	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	57.37	0.19	19.25
3	-9	N/A	N/A	N/A	N/A	N/A	N/A	229.31	0.73	73.24	38.29	0.13	13.26
4	-7	N/A	N/A	N/A	N/A	N/A	N/A	76.47	0.25	25.25	28.76	0.10	10.27
5	-5	N/A	N/A	N/A	N/A	N/A	N/A	45.92	0.16	15.66	23.05	0.08	8.48
6	-3	N/A	N/A	N/A	114.67	0.37	37.25	32.84	0.12	11.55	19.25	0.07	7.28
7	-1	N/A	N/A	N/A	57.37	0.19	19.25	25.58	0.09	9.27	16.54	0.06	6.43
8	1	229.31	0.73	73.24	38.29	0.13	13.26	20.97	0.08	7.83	14.52	0.06	5.80
9	3	76.47	0.25	25.25	28.76	0.10	10.27	17.79	0.07	6.83	12.95	0.05	5.31
10	5	45.92	0.16	15.66	23.05	0.08	8.48	15.46	0.06	6.10	11.70	0.05	4.91
11	7	32.84	0.12	11.55	19.25	0.07	7.28	13.69	0.06	5.54	10.68	0.05	4.59
12	9	25.58	0.09	9.27	16.54	0.06	6.43	12.29	0.05	5.10	9.84	0.04	4.33
13	11	20.97	0.08	7.83	14.52	0.06	5.80	11.17	0.05	4.75	9.13	0.04	4.11
14	13	17.79	0.07	6.83	12.95	0.05	5.31	10.24	0.04	4.46	8.52	0.04	3.92
15	15	15.46	0.06	6.10	11.70	0.05	4.91	9.47	0.04	4.21	8.00	0.04	3.75

Bibliography

[1] *Clearances and conductor spacings*. Transpower. 2009. URL: https://www.ea.govt.nz/dmsdocument/4121.

[2] NZ STANDARDS. *AS NZS 7000:2016 - Overhead line design*. 2016.

[3] Warren Wang and Sarah Pinter. *Dynamic Line Rating Systems for Transmission Lines*. Tech. rep. U.S. Department of Energy, 2014. URL: https://www.smartgrid.gov/files/SGDP_Transmission_DLR_Topical_Report_04-25-14_FINAL.pdf.

[4] Dalibor Kladar. *Dynamic Line Rating in the world - Overview*. Tech. rep. XpertPower Associates Ltd, 2014. URL: https://www.researchgate.net/publication/260229962_Dynamic_Line_Rating_in_the_world_-_Overview.

[5] Adonis Dino and Angus Ketley. *Dynamic Transmission Line Rating Technology Review*. Tech. rep. Electricity Commission of New Zealand, 2009. URL: https://www.ea.govt.nz/dmsdocument/1872.

[6] Chris Mensah-Bonsu. "Instrumentation and measurement of overhead conductor sag using the differential global positioning satellite system". PhD thesis. Arizona State University, 2000.

[7] *Innovative solution for Dynamic Line Rating*. Ampacimon. 2015. URL: http://www.ampacimon.com/wp-content/uploads/2015/09/Ampacimon-Brochure.pdf.

[8] *Dynamic Line Rating for overhead lines – V6*. ENTSO-E. 2015. URL: https://docstore.entsoe.eu/Documents/SOC%20documents/Regional_Groups_Continental_Europe/Dynamic_Line_Rating_V6.pdf.

[9] Mauricio Müller, João Hoffmann, and Ana Kersting. "Transmission Line Up-rating Design Using Survey Data from Airborne Lidar". In: *Cigré*. Aug. 2006.

[10] E. Golinelli, U. Perini, and G. Ogliari. "A new IR laser scanning system for power lines sag measurements". In: *18th Italian National Conference on Photonic Technologies (Fotonica 2016)*. Institution of Engineering and Technology, 2016. DOI: 10.1049/cp.2016.0919. URL: https://doi.org/10.1049%2Fcp.2016.0919.

[11] *Cable Distance Meter*. HD Electric Company. 2004. URL: http://www.hdelectriccompany.com/cdm_lit.pdf.

[12] Josh McCulloch and Richard Green. "Extraction of utility poles in LIDAR scans using cross-sectional slices". In: *2016 International Conference on Image and Vision Computing New Zealand (IVCNZ)*. IEEE, Nov. 2016. DOI: 10.1109/ivcnz.2016.7804442. URL: https://doi.org/10.1109%2Fivcnz.2016.7804442.

[13] Josh McCulloch and Richard Green. "Utility pole extraction using vehicle-mounted LIDAR for dynamic line rating". In: *2017 International Conference on Image and Vision Computing New Zealand (IVCNZ)*. IEEE, Dec. 2017. DOI: 10.1109/ivcnz.2017.8402484. URL: https://doi.org/10.1109%2Fivcnz.2017.8402484.

[14] Josh McCulloch and Richard Green. "Density based Recovery of Urban Power Lines using Vehicle-Mounted LiDAR". In: *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)*. IEEE, Nov. 2018.

[15] Mehran Keshavarzian and Charles H Priebe. "Sag and tension calculations for overhead transmission lines at high temperatures-modified ruling span method". In: *IEEE Transactions on Power Delivery* 15.2 (2000), pp. 777–783.

[16] ML Lu, G Pfrimmer, and Z Kieloch. "Upgrading an existing 138 kV transmission line in Manitoba". In: *Power Engineering Society General Meeting, 2006. IEEE*. IEEE. 2006, 6–pp.

[17] R. Valerie Ussyshkin et al. "Advantages of Airborne Lidar Technology in Power Line Asset Management". In: *2011 International Workshop on Multi-Platform/Multi-Sensor Remote Sensing and Mapping*. IEEE,

Jan. 2011. DOI: 10.1109/m2rsm.2011.5697427. URL: https://doi.org/10.1109%2Fm2rsm.2011.5697427.

[18] Shu-ichi Ashidate, Susumu Murashima, and Noritsuna Fujii. "Development of a helicopter-mounted eye-safe laser radar system for distance measurement between power transmission lines and nearby trees". In: *IEEE transactions on power delivery* 17.2 (2002), pp. 644–648.

[19] Michael Frank et al. "Vegetation management of utility corridors using high-resolution hyperspectral imaging and LiDAR". In: *2010 2nd Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing*. IEEE, June 2010. DOI: 10.1109/whispers.2010.5594887. URL: https://doi.org/10.1109%2Fwhispers.2010.5594887.

[20] Steven J. Mills et al. "Evaluation of Aerial Remote Sensing Techniques for Vegetation Management in Power-Line Corridors". In: *IEEE Transactions on Geoscience and Remote Sensing* 48.9 (Sept. 2010), pp. 3379–3390. DOI: 10.1109/tgrs.2010.2046905. URL: https://doi.org/10.1109%2Ftgrs.2010.2046905.

[21] Simon Clode and Franz Rottensteiner. "Classification of Trees and Powerlines from medium resolution Airborne Laserscanner data in Urban Environments". In: *APRS Workshop on Digital Image Computing*. 2005.

[22] Yoonseok Jwa and Gunho Sohn. "A Piecewise Catenary Curve Model Growing for 3D Power Line Reconstruction". In: *Photogrammetric Engineering & Remote Sensing* 78.12 (Dec. 2012), pp. 1227–1240. DOI: 10.14358/pers.78.11.1227. URL: https://doi.org/10.14358%2Fpers.78.11.1227.

[23] Lingli Zhu and Juha Hyyppä. "Fully-Automated Power Line Extraction from Airborne Laser Scanning Point Clouds in Forest Areas". In: *Remote Sensing* 6.11 (Nov. 2014), pp. 11267–11282. DOI: 10.3390/rs61111267. URL: https://doi.org/10.3390%5C%2Frs61111267.

[24] G. Sohn, Y. Jwa, and H. B. Kim. "Automatic Powerline Scene Classification and Reconstruction Using Airborne Lidar Data". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* I-3 (July 2012), pp. 167–172. DOI: 10.5194/isprsannals-i-3-167-2012. URL: https://doi.org/10.5194%2Fisprsannals-i-3-167-2012.

[25] Alena Otcenasova, Marek Hoger, and Juraj Altus. "Possible use of airborne LiDAR for monitoring of power lines in Slovak Republic". In: *Proceedings of the 2014 15th International Scientific Conference on Electric Power Engineering (EPE)*. IEEE, May 2014. DOI: 10.1109/epe.2014.6839504. URL: https://doi.org/10.1109%2Fepe.2014.6839504.

[26] R.A. Mclaughlin. "Extracting Transmission Lines From Airborne LIDAR Data". In: *IEEE Geoscience and Remote Sensing Letters* 3.2 (Apr. 2006), pp. 222–226. DOI: 10.1109/lgrs.2005.863390. URL: https://doi.org/10.1109%2Flgrs.2005.863390.

[27] Y Jwa, G Sohn, and H B Kim. "Automatic 3D Powerline Reconstruction Using Airborne Lidar Data". In: *Bretar F, Pierrot-Deseilligny M, Vosselman G (Eds) Laser scanning 2009, IAPRS* XXXVIII-3/W8.2004 (Sept. 2009), pp. 105–110.

[28] Bo Guo et al. "An Improved Method for Power-Line Reconstruction from Point Cloud Data". In: *Remote Sensing* 8.1 (Jan. 2016), p. 36. DOI: 10.3390/rs8010036. URL: https://doi.org/10.3390%2Frs8010036.

[29] Liang Cheng et al. "Extraction of Urban Power Lines from Vehicle-Borne LiDAR Data". In: *Remote Sensing* 6.4 (Apr. 2014), pp. 3302–3320. DOI: 10.3390/rs6043302. URL: https://doi.org/10.3390%2Frs6043302.

[30] Jing Liang et al. "A New Power-Line Extraction Method Based on Airborne LiDAR Point Cloud Data". In: *2011 International Symposium on Image and Data Fusion*. IEEE, Aug. 2011. DOI: 10.1109/isidf.2011.6024293. URL: https://doi.org/10.1109%2Fisidf.2011.6024293.

[31] Thomas Melzer and Christian Briese. “Extraction and Modeling of Power Lines from ALS Point Clouds”. In: (2004).

[32] Gunho Sohn and Ian Dowman. “Data fusion of high-resolution satellite imagery and LiDAR data for automatic building extraction”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 62.1 (2007), pp. 43–63.

[33] Tee-Ann Teo and Chi-Min Chiu. “Pole-Like Road Object Detection From Mobile Lidar System Using a Coarse-to-Fine Approach”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 8.10 (Oct. 2015), pp. 4805–4818. DOI: 10.1109/jstars.2015.2467160. URL: https://doi.org/10.1109%2Fjstars.2015.2467160.

[34] C Cabo et al. “An algorithm for automatic detection of pole-like street furniture objects from Mobile Laser Scanner point clouds”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 87 (2014), pp. 47–56.

[35] A Bienert et al. “Tree detection and diameter estimations by analysis of forest terrestrial laserscanner point clouds”. In: *ISPRS workshop on laser scanning*. Vol. 2007. 2007, pp. 50–55.

[36] J. Demantké et al. “Dimensionality Based Scale Selection in 3d Lidar Point Clouds”. In: *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XXXVIII-5/W12 (Sept. 2012), pp. 97–102. DOI: 10.5194/isprsarchives-xxxviii-5-w12-97-2011. URL: https://doi.org/10.5194%2Fisprsarchives-xxxviii-5-w12-97-2011.

[37] Sylvie Soudarissanane et al. “Incidence Angle Influence on the Quality of Terrestrial Laser Scanning Points”. In: *ISPRS Workshop Laserscanning 2009* XXXVIII-3/W8 (2009).

[38] De-an Luo and Yan-min Wang. “Rapid extracting pillars by slicing point clouds”. In: *Proc. XXI ISPRS Congress, IAPRS*. Vol. 37. Citeseer. 2008, pp. 215–218.

[39] Matti Lehtomäki et al. “Detection of vertical pole-like objects in a road environment using vehicle-based laser scanning data”. In: *Remote Sensing* 2.3 (2010), pp. 641–664.

[40] Clément Mallet, Frederic Bretar, and Uwe Soergel. “Analysis of full-waveform lidar data for classification of urban areas”. In: *Photogrammetrie Fernerkundung Geoinformation* 5 (2008), pp. 337–349.

[41] Jens Behley, Volker Steinhage, and Armin B Cremers. “Performance of histogram descriptors for the classification of 3d laser range data in urban environments”. In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE. 2012, pp. 4391–4398.

[42] P Press, David Austin, et al. “Approaches to pole detection using ranged laser data”. In: *Proceedings of Australasian Conference on Robotics and Automation*. Citeseer. 2004.

[43] Jing Huang and Suya You. “Pole-like object detection and classification from urban point clouds”. In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 3032–3038.

[44] Shi Pu et al. “Recognizing basic structures from mobile laser scanning data for road inventory studies”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 66.6 (2011), S28–S39.

[45] Aleksey Golovinskiy, Vladimir G Kim, and Thomas Funkhouser. “Shape-based recognition of 3D point clouds in urban environments”. In: *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE. 2009, pp. 2154–2161.

[46] Yongtao Yu et al. “Semiautomated extraction of street light poles from mobile LiDAR point-clouds”. In: *IEEE Transactions on Geoscience and Remote Sensing* 53.3 (2015), pp. 1374–1386.

[47] H Yokoyama et al. “Detection and classification of pole-like objects from mobile laser scanning data of urban environments”. In: *Int. J. CAD/CAM* 13 (Jan. 2013), pp. 31–40.

[48] Federico Tombari et al. “Automatic detection of pole-like structures in 3d urban environments”. In: *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE. 2014, pp. 4922–4929.

[49] Jorge Hernández and Beatriz Marcotegui. “Point cloud segmentation towards urban ground modeling”. In: *Urban Remote Sensing Event, 2009 Joint*. IEEE. 2009, pp. 1–5.

[50] Andrés Serna and Beatriz Marcotegui. “Detection, segmentation and classification of 3D urban objects using mathematical morphology and supervised learning”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 93 (July 2014), pp. 243–255. DOI: 10.1016/j.isprsjprs.2014.03.015. URL: https://doi.org/10.1016%2Fj.isprsjprs.2014.03.015.

[51] Alexander Velizhev, Roman Shapovalov, and Konrad Schindler. “Implicit shape models for object detection in 3D point clouds”. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 3 (2012), pp. 179–184.

[52] *New Zealand Electrical Code of Practice for Electrical Safe Distances*. Tech. rep. Ministry of Consumer Affairs, 2001. URL: https://www.transpower.co.nz/sites/default/files/publications/resources/NZECP%2034%202001%20-%20New%20Zealand%20Electrical%20Code%20of%20Practice%20for%20Electrical%20Safe%20Distances%20Published%2021%20December%202001.pdf.

[53] Martin LaMonica. *Microsoft’s Kinect: A robot’s low-cost, secret weapon*. 2011. URL: https://www.cnet.com/news/microsofts-kinect-a-robots-low-cost-secret-weapon/.

[54] *Intel RealSense Camera R200*. Intel. 2016. URL: https://software.intel.com/sites/default/files/managed/d7/a9/realsense-camera-r200-product-datasheet.pdf.

[55] *Intel RealSense Camera SR300*. Intel. 2016. URL: http://www.mouser.com/ds/2/612/realsense-sr300-product-datasheet-rev-1-0-994604.pdf.

[56] Hamed Sarbolandi, Damien Lefloch, and Andreas Kolb. “Kinect range sensing: Structured-light versus Time-of-Flight Kinect”. In: *Computer vision and image understanding* 139 (2015), pp. 1–20.

[57] Péter Fankhauser et al. “Kinect v2 for mobile robot navigation: Evaluation and modeling”. In: *2015 International Conference on Advanced Robotics (ICAR)*. IEEE. 2015, pp. 388–394.

[58] Arthur P Cracknell. *Introduction to remote sensing*. CRC press, 2007.

[59] *Velodyne VLP-16*. Velodyne. 2017. URL: http://velodynelidar.com/docs/datasheet/63-9229_Rev-E_Puck%20_Spec%20Sheet_Web.pdf.

[60] Morgan Quigley et al. “ROS: an open-source Robot Operating System”. In: *ICRA workshop on open source software*. Vol. 3. 3.2. Kobe, Japan. 2009, p. 5.

[61] Ji Zhang and Sanjiv Singh. “LOAM: Lidar Odometry and Mapping in Real-time”. In: *Robotics: Science and Systems X*. Robotics: Science and Systems Foundation, July 2014. DOI: 10.15607/rss.2014.x.007. URL: https://doi.org/10.15607%2Frss.2014.x.007.

[62] Michael G Wing, Aaron Eklund, and Loren D Kellogg. “Consumer-grade global positioning system (GPS) accuracy and reliability”. In: *Journal of forestry* 103.4 (2005), p. 169.

[63] Eric Royer et al. “Localization in urban environments: monocular vision compared to a differential GPS sensor”. In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 2. IEEE. 2005, pp. 114–121.

[64] T Aschoff and H Spiecker. “Algorithms for the automatic detection of trees in laser scanner data”. In: *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 36.Part 8 (2004), W2.

[65] Jaromĭr Landa, David Procházka, and Jiřĭ Šťastný. “Point cloud processing for smart systems”. In: *Acta Universitatis Agriculturae et Silviculturae Mendelianae Brunensis* 61.7 (2013), pp. 2415–2421. DOI: 10.11118/actaun201361072415. URL: https://doi.org/10.11118%2Factaun201361072415.

[66] Domen Mongus and Borut Žalik. “Parameter-free ground filtering of LiDAR data for automatic DTM generation”. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 67 (2012), pp. 1–12.

[67] Anttoni Jaakkola et al. “Retrieval algorithms for road surface modelling using laser-based mobile mapping”. In: *Sensors* 8.9 (2008), pp. 5238–5249.

[68] Matthew Carlberg et al. “Classifying urban landscape in aerial LiDAR using 3D shape analysis”. In: *Image Processing (ICIP), 2009 16th IEEE International Conference on*. IEEE. 2009, pp. 1701–1704.

[69] Dave Kelbe et al. “Reconstruction of 3D tree stem models from low-cost terrestrial laser scanner data”. In: *Laser Radar Technology and Applications XVIII*. Ed. by Monte D. Turner and Gary W. Kamerman. SPIE, May 2013. DOI: 10.1117/12.2015963. URL: https://doi.org/10.1117%2F12.2015963.

[70] Jean-François Lalonde et al. “Natural terrain classification using three-dimensional ladar data for ground robot mobility”. In: *Journal of field robotics* 23.10 (2006), pp. 839–861.

[71] K. Fukunaga and L. Hostetler. “The estimation of the gradient of a density function, with applications in pattern recognition”. In: *IEEE Transactions on Information Theory* 21.1 (Jan. 1975), pp. 32–40. DOI: 10.1109/tit.1975.1055330. URL: https://doi.org/10.1109%2Ftit.1975.1055330.

[72] Sameer Agarwal, Keir Mierle, et al. *Ceres Solver*. http://ceres-solver.org. 2010.

[73] David Arthur and Sergei Vassilvitskii. “k-means++: The advantages of careful seeding”. In: *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics. 2007, pp. 1027–1035.